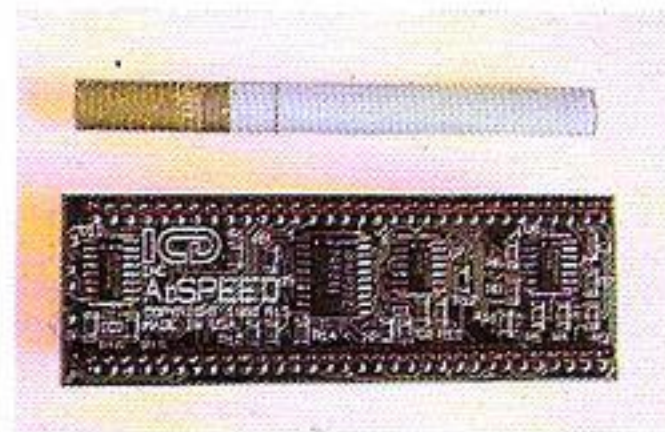
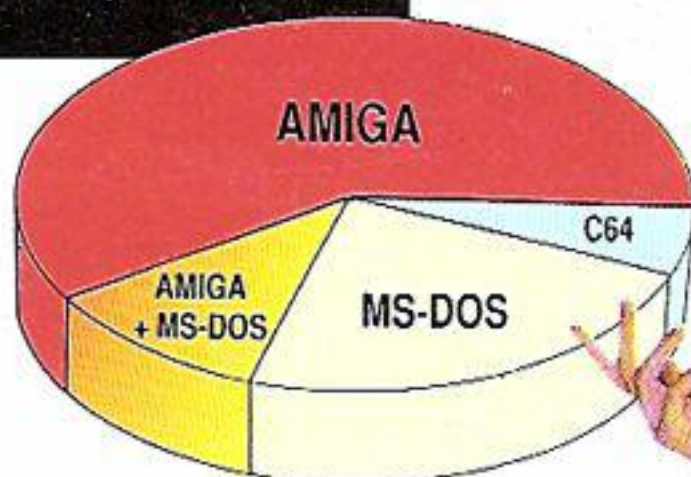
Anno XI - N. 87 - 25 agosto-25 settembre 1991 - Sped. Abb. Post. Gr III/70 - CR - Distr.: Parrini

A 500

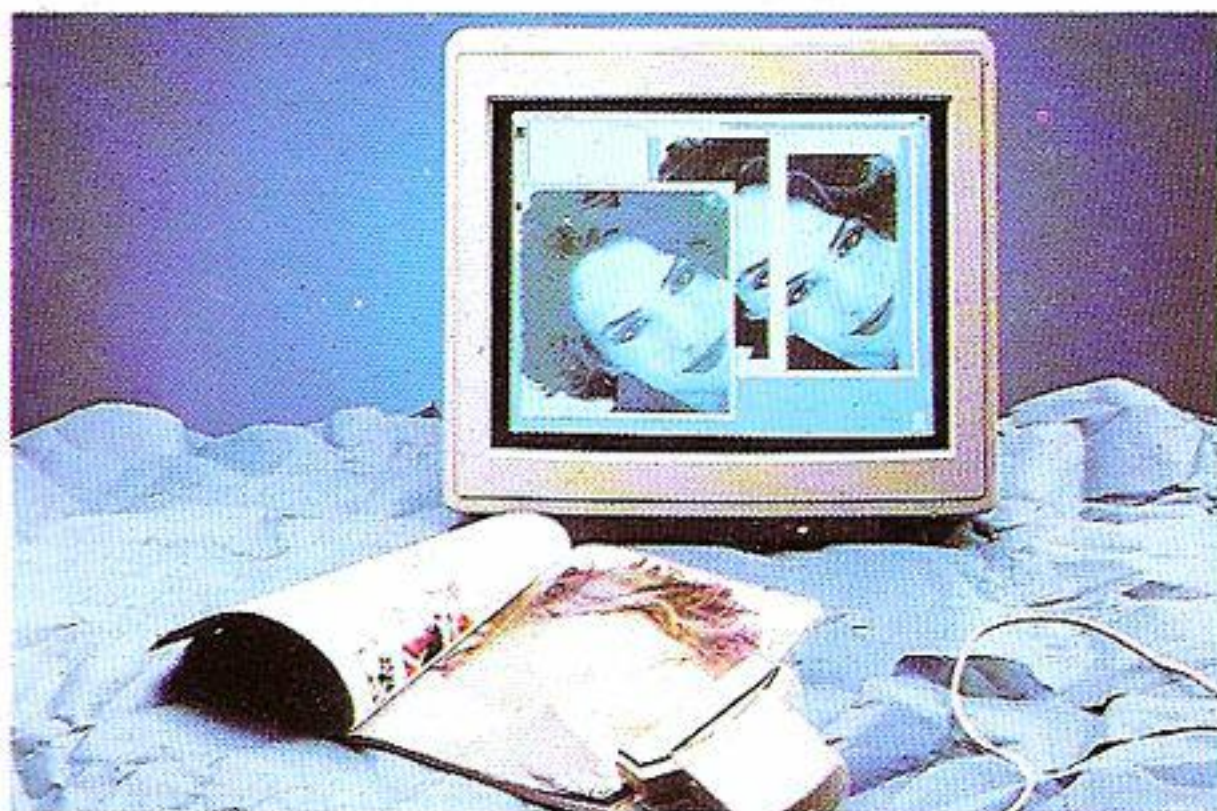
Minischeda acceleratrice

INCHIESTA

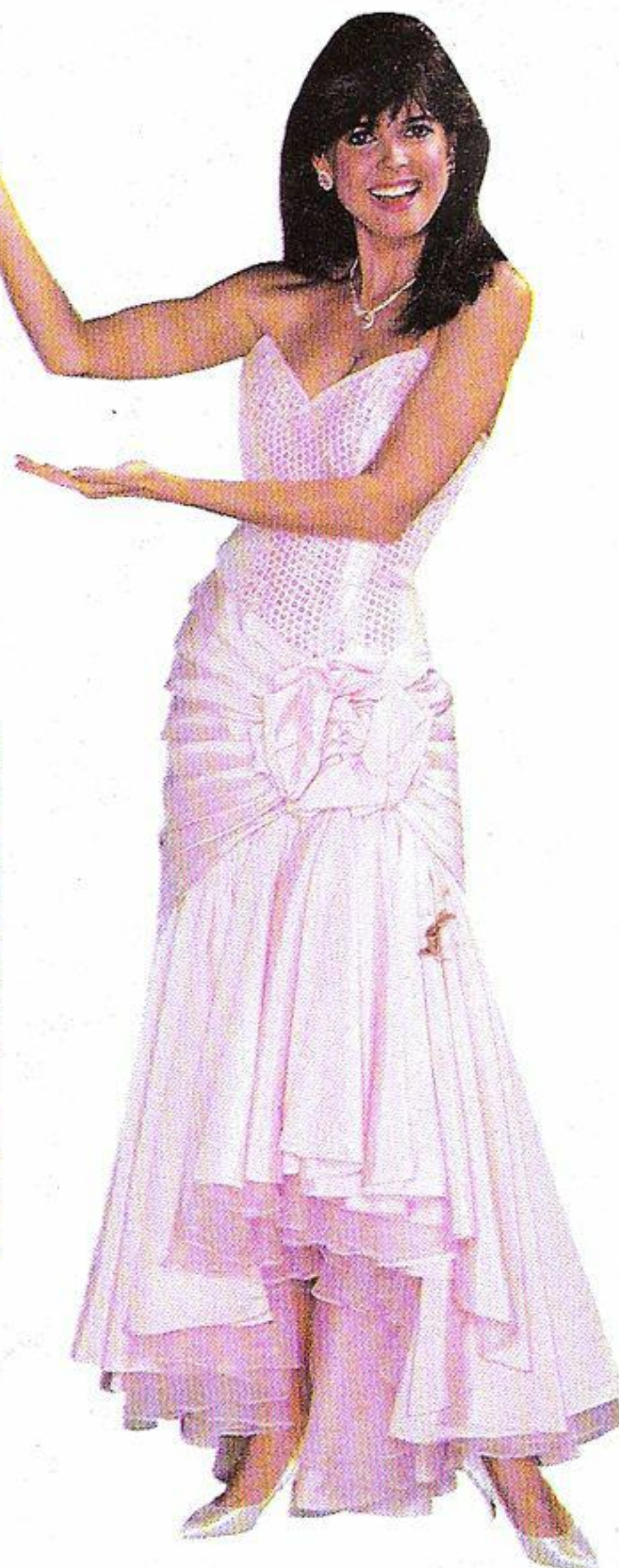
Ecco, come siamo

AMIGA
C64
AMIGA + MS-DOS
MS-DOS

AMIGA

- Solidi in rotazione
- Grafica in Kick-Pascal
- Assembly: gli indirizzamenti

STAMPANTI

Dalla carta al video e ritorno

Briscola con C64, Amiga ed Ms-Dos

La vostra posta • I libri di Ventura • Emulatore di virus • La sfida di Pitagora • Contatempo in Turbo Pascal • Assembly 80X86 • Enciclopedia Ms-Dos • Amiga C • Inchiesta sulle stampanti • La tua foto nel computer

Sommario

C/64, Amiga, Ms - Dos

25 Due giochi per C/64, Amiga Ms - Dos
Lo scarso numero di lettori che usano ancora il vecchio computer ci costringe a trattare un argomento che possa soddisfare anche le altre due categorie di lettori: gli **Amighi** e gli **Ms- Dossiani**

Amiga

14 Piagora, il suo teorema e gli sfidanti
E' la risposta, di tre "Amighi", alla sfida lanciata sul n. 85.

54 Amiga a tutta birra
Due nuove schede acceleratrici, la prima economica, la seconda un po' meno...

59 Solidi in rotazione prospettica
Un breve programma in AmigaBasic consente di visualizzare solidi in prospettiva mentre effettuano rotazioni attorno ai tre assi X, Y, Z.

65 PostAmiga
Consueto appuntamento mensile con le vostre domande e i vostri dubbi.

76 Tutti gli indirizzamenti del 68000
Come il processore di Amiga individua una locazione di memoria, tra le tante, utilizzando i vari metodi di puntamento.

80 Amiga si affaccia alla finestra
Come realizzare finestre e schermi operando con il linguaggio C.

Mondo Ms - Dos

22 Un contatempo in T. Pascal
Un programma più breve ed efficiente di quello pubblicato sul numero scorso.

32 Radiografiamo il PC
Uno stesso algoritmo, sviluppato in Basic, C, T. Pascal, per determinare la configurazione del nostro Ms - Dos compatibile.

39 Te la do io la directory!
Procedura, in puro Assembly, per visualizzare il contenuto dei vostri dischetti.

Amiga + Ms - Dos

4 Editoriale

5 La vostra posta

7 Libro e dischetto, utente perfetto
Come dovrebbero essere scritti i volumi acquistabili in libreria? Un esempio con **Ventura Publisher.**

12 Come ti acchiappo il virus
Un suggerimento innocuo per sperimentare l'efficacia dei programmi Anti-Virus.

20 Diamoci un taglio!
E' la consueta sfida mensile. Mettete alla prova la vostra bravura di programmatori.

43 Audio, Video, Mouse e Copiatori
Alcuni accessori hardware sono disponibili sia nella versione Amiga che Ms - Dos.

52 Metti una foto nel tuo computer
Un accessorio di sicuro interesse che vedremo diffondersi presso gli appassionati... abbastanza ricchi!

69 Miglioriamo la grafica in Pascal
Come creare una Unit grafica, in **KickPascal Amiga**, compatibile con il più prestigioso **Turbo Pascal Borland**, il noto compilatore operante in ambiente Ms - Dos.

88 Come si stampa oggi e perchè
Una breve introduzione sul mondo della editoria professionale, ma non spaventatevi: potete realizzare anche voi impaginazioni perfette!

92 Iniziamo con la stampante
E' la breve premessa di una serie di articoli dedicati alla indispensabile periferica.

94 I risultati dell'inchiesta.
Chi è il lettore "medio"di Computer Club? Vediamo di scoprirlo

95 Inchiesta sulle stampanti

COMPUTER CLUB

Direttore: Alessandro de Simone

Coordinatore: Marco Miotti

Redazione / Collaboratori:
Davide Ardizzone - Claudio Baiocchi
Luigi Callegari - Umbero Colapicchioni
Donato De Luca - Carlo d'Ippolito
Valerio Ferri - Michele Maggi
Giancarlo Mariani - Domenico Pavone
Armando Sforzi - Dario Pistella
Fabio Sorgato- Valentino Spataro
Franco Rodella- Stefano Simonelli
Luca Viola

Direzione:
Via Mosè, 22 cap. 20090 OPERA (Mi)

Telefono 02 / 57.60.63.10
Fax 02 / 57.60.30.39
BBS 02 / 57.60.52.11

Pubblicità:
Leandro Nencioni (dir. vendite)
Via Mose', 22 20090 Opera (Mi)
tel. 02 / 57.60.63.10

Emilia Romagna:
Spazio E
P.zza Roosvelt, 4 cap. 40123 Bologna
Tel. 051 /23.69.79

Toscana, Marche, Umbria
Mercurio s.r.l. Via Rodari, 9
S. G.nni Valdarno (Ar)
Tel. 055 / 94.74.44

Lazio, Campania
Spazio Nuovo
Via P. Foscari, 70
cap. 00139 Roma
tel. 06 / 81.09.679

Abbonamenti: Liliana Spina
Arretrati e s/w: Lucia Dominoni

Tariffe: Prezzo per copia L. 6000
Abbonamento annuo (11 fascicoli) L. 60000
Estero: L. 100000 - Indirizzare versamenti a:
Systems Editoriale Srl
c/c 37952207 oppure inviare comune assegno bancario non trasferibile e barrato due volte a:
Systems Editoriale Srl (servizio arretrati)
Via Mose', 22
cap. 20090 OPERA (Mi)

Composizione: Systems Editoriale
La Litografica Srl Cuggiono (Mi)

Registrazioni: Tribunale di Milano
n. 370 del 2/10/82

Direttore Responsabile: Michele Di Pisa

Spedizioni in abbonamento postale gruppo III.
Pubblicita' inferiore al 70%

Distributore: Parrini - Milano

Periodici Systems:
Amiga Club (disco ed. Germania) - Banca Oggi - Computer (quotidiano) - Computer Club - 64 Club (disco) - Computer Club (disco ed. Germania) - Hospital Management - Nursing '90 - PC Club (disco ed. Germania) - Personal Computer - Jonathan - VR - Videoteca

I risultati dell'inchiesta

Non è certo la prima volta che viene individuato, con il mese di settembre, il periodo tradizionale da cui partono iniziative nuove, se ne rafforzano altre e si riflette su tutte le decisioni prese in precedenza.

Capitano a proposito i risultati dell'inchiesta, condotta sui numeri 85 e 86, che ci consentono di "aggiustare il tiro" sulla direzione che la nostra rivista deve seguire.

Che l'Amiga fosse il computer più amato, l'avevamo capito dalle lettere che giungevano in Redazione, tanto da costringerci ad aprire una rubrica fissa, PostAmiga, tenuta dall'ottimo Domenico Pavone.

Ma che il sistema MS - DOS avesse un elevato numero di proseliti, sinceramente, non ce l'aspettavamo; nè pensavamo che i 64-isti costituissero, ormai, una minoranza tanto esigua.

Le configurazioni in possesso dai nostri lettori (chi si aspettava un 10% di Amighi con Hard Disk ed oltre il 50% di possessori di stampanti a 24 aghi?) sono di tutto rispetto e la richiesta di sfruttare al massimo i loro sistemi conferma il dato estremamente positivo.

Al momento di scrivere queste note non è ancora stata presa una decisione che, a livello di iniziative didattiche, dovrebbe costituire una piacevole sorpresa per il 95% (almeno...) dei nostri lettori; tenete bene aperti gli occhi, comunque, quando vi recate in edicola: cercate di individuare, tra le riviste del settore (su disco e su carta) eventuali novità *targate* **Systems**. Per ora non ci è dato dire di più di quanto accennato nelle rubriche della Posta.

Per ciò che riguarda il fascicolo che avete tra le mani, invece, sono da segnalare vari, ottimi programmi.

Il primo di questi, per sistemi MS - DOS, consente di determinare con precisione, e con la minima fatica, il tempo che passiamo in compagnia di vari programmi: installatelo permanentemente sul vostro hard disk (ormai sappiamo che ne avete uno...) ed inizializzatelo con tutti i programmi che usate frequentemente.

Tra i listati Amiga, invece, c'è una scelta maggiore, e non poteva essere altrimenti: si va dalla soluzione del teorema di Pitagora alla rotazione, in prospettiva, di solidi geometrici; dalla gestione delle finestre in C alla creazione di Unit grafiche in Kick Pascal.

Per ciò che riguarda le novità hardware, poi, non c'è che l'imbarazzo della scelta, tra schede di vario tipo, tutte interessanti.

Il tutto, ovviamente, per andare incontro alle esigenze dei nostri lettori, che ringraziamo per aver risposto alla nostra inchiesta e che, ne siamo sicuri, ci aiuteranno sempre più, con le prossime, a soddisfare le loro richieste.

Sempre nel loro esclusivo interesse, ovvio!

Alessandro de Simone

LA VOSTRA POSTA

(a cura di A. de Simone)

Il PC nell'Amiga

Quali sono le velocità effettive disponibili sulle varie schede MS - DOS compatibili (ATonce e simili) e quali sono i programmi professionali che possono girarvi?
(Ugo S. - Castiglione)

La domanda, forse posta in modo un po' ingenuo, fa capire che ci si illude ancora di avere a disposizione, su Amiga (e per di più sul modello **A-500**) un *completo* elaboratore MS - DOS *compatibile*.

Per quanto riguarda la compatibilità di base, non c'è dubbio che le schede emulatrici svolgono abbastanza bene il loro compito.

Per ciò che riguarda la possibilità di emulare le varie schede grafiche, invece, ci potrebbe essere qualche difficoltà a causa dell'inalterabilità della circuiteria elettronica montata su Amiga.

In ogni caso, comunque, la possibilità di avere a disposizione un **vero** computer MS - DOS incontra insormontabili ostacoli nella necessità, ormai assoluta, di disporre di un disco rigido, periferica notoriamente poco diffusa tra la gran massa degli utenti Amiga (vedi risultati dell'inchiesta su questo stesso numero).

Il lettore pensi che il sottoscritto è stato "costretto" a passare ad un computer **80386** DX (**25** Mhz) con disco rigido da **80** megabyte a causa della lentezza del precedente computer **80286** (**12** Mhz) dotato di hard disk di *soli* **20** megabyte!

Se, fino a qualche anno fa, la configurazione accennata poteva esser sufficiente, la indiscutibile esigenza di avere a disposizione **grandi** quantità di spazi su hard disk, richiesta da **tutti** i programmi professionali, relega i computer che ne sono privi in angoli di scarsa utilità pratica. In conclusione: se volete operare seriamente in ambiente MS - DOS compratevi un computer di questo standard (ormai i prezzi sono *quasi* irrisori se confrontati alle schede per Amiga *affiancate* da hard disk).

Se, invece, volete limitarvi a curiosare, una scheda emulatrice può esser sufficiente.

❏ ❐

Uso sofisticato del C/64

Nei pochi numeri che ci separano dalla fine del 1991 (data a partire dalla quale avete intenzione di non affrontare più problemi relativi al C/64) potreste almeno parlare di programmazione avanzata (realizzazione di schermate multicolor, uso di compilatori e decompilatori, eccetera). Come si potrebbe, altrimenti, passare a sistemi superiori se non si conosce a fondo il C/64?
(Antonio Barbuto - Salerno)

Si può, si può. Anzi, è molto più facile con i sistemi superiori che con il C/64.

Purtroppo il nostro lettore, come tanti altri che desiderano usare ancora il C/64, non possono rendersi conto delle notevoli potenzialità offerte da un **Amiga** oppure da un **MS - DOS**. Se fosse loro possibile passare *un solo pomeriggio* in compagnia di un amico dotato di uno di tali computer, cambierebbero certamente idea.

E' tutt'altra cosa, credetemi.

Del resto, se non mi credete, prima o poi sarete costretti

Un'altra voce (vedi C.C. n. 85 pag. 6)

Con questa mia voglio rispondere alla lettera di **Mauro Bossetti**, pubblicata su C.C. n. 85, che denigra i sistemi MS - DOS a favore di **Amiga** e contesta il modo con cui la rivista tratta quest'ultimo computer.

Innanzitutto non mi sembra che, negli ultimi numeri, si stia parlando male di Amiga: basta contare le pagine ad esso dedicate per accorgersi che il sistema veramente maltrattato sia l'MS - DOS.

Per quanto riguarda la **grafica**, sono completamente in disaccordo con il signor Bossetti. In un primo tempo egli afferma che la grafica Dossiana riesce ad umiliare Amiga. Poi, per aver ragione, paragona la modalità Ham alla VGA standard. Beh, se vogliamo parlare di metodi di visualizzazione non ufficiali, ricordo solo che anche con una semplice VGA 800 x 600 e 512 Kram (ne ho trovata una *sotto* il centinaio di migliaio di lire) si possono visualizzare contemporaneamente tutti i 262144 colori che la scheda può generare.

Se, poi, si tira in ballo il discorso BCS, consiglio di chiedere, in un qualsiasi negozio realmente specializzato, di osservare una schermata generata da AutoCAD ed ombreggiata da Autosade su un sistema IBM 8514a dotato di 16 milioni di colori e 1280 x 1024 pixel di risoluzione...

Per ciò che riguarda il **suono**, devo dare ragione: quello standard fa veramente pietà tanto che ho disattivato l'altoparlante del mio sistema direttamente con forbice e nastro isolante. Ma non mi sembra il caso di tirare in ballo la scheda SoundBlaster; sarei più propenso ad orientarmi verso qualcosa di più professionale. Non so, magari un sistema MS - DOS interfacciato con due hard disk da 800 mega per registrare in tempo reale un intero disco da elaborare e riversare, poi, direttamente su CD con un Worm Sony dell'ultima generazione. Le case discografiche non usano certo l'Amiga per realizzare master *professionali*.

Non riesco ad apprezzare quella gran velocità(!) di Amiga-DOS. Proprio in uno di questi giorni ho avuto modo di lavorare contemporaneamente su un sistema **8086** (8 Mhz) ed un **Amiga 500** dotato di solo drive interno.

Beh, se devo dire la verità, abituato a lavorare con Windows 3.0 e poiché non mi piace il sadomasochismo, ho aperto lo Shell ed ho gettato via il Workbench perché mi sembrava di avere sotto gli occhi il GEOS di un C/64 con clock ridotto a 0.05 Mhz.

Per quanto riguarda i linguaggi, non mi sembra possibile paragonare il GFA Basic (grazie, Atari...) con il QuickBasic. Il primo è stato sviluppato esclusivamente per ottenere grandi velocità a discapito di altre funzioni, mentre non è così per il secondo. Se ciò fosse, la MicroSoft si pesterebbe da sola i piedi in quanto venderebbe meno pacchetti Quick Pascal e QuickC.

Un ultimo consiglio: non paragonate più il nuovo **Workbench 2.0** con **Windows 3.0**. Se volete confrontare seriamente, prendete un **Amiga 3000** a 25 Mhz con 120 megabyte di hard disk e paragonatelo con un *qualsiasi* made in **Taiwan 80486** a 25 Mhz con 120 mega di H/d. Aspettate che Amiga 3000 si affermi (campa cavallo) e che esca la nuova versione Windows 4.0 dotato di Dos extension per far funzionare anche i programmi Dos in Protect Mode; poi vediamo chi vince la sfida.

Francesco Gringoli - Padova

ad usare un Amiga oppure un MS - DOS dal momento che li troverete nei fustini di detersivo o nei pacchetti di patatine (magari formato famiglia...).

Come è già successo per il **C/64**, ai tempi in cui era in auge il glorioso **Vic 20**!

Carta riciclata

Mi fa piacere che anche voi, come tante altre case editrici, stampiate la rivista su carta riciclata. Ritengo che sia un gesto intelligente che consente di risparmiar la vita di tanti alberi...
(da alcune lettere)
Perché non cambiate tipo di carta, che puzza e fa pure schifo?
(da una felice minoranza di lettori)

Viviamo in un mondo in cui **apparire** vale, purtroppo, molto di più che **essere**. Lo dimostrano varie tendenze e mode, che invitano allo spreco più sfrenato, trascurando esigenze spesso di basilare importanza.

Per fortuna ci rivolgiamo ad un'utenza che bada più a ciò che scriviamo che alla carta su cui lo facciamo.

Per onore della verità, comunque, devo ammettere che la decisione di usare carta riciclata deriva, oltre che da convinzioni ecologiche, da spudorate motivazioni economiche.

Quelle stesse motivazioni che però consentono, a coloro cui non piace la carta riciclata, di acquistare la nostra rivista ad un prezzo di copertina di gran lunga più modesto di quanto sarebbe se decidessimo di pubblicare su carta patinata in quadricromia.

C++, chi lo vuole?

Protesto perché C.C. non tratta argomenti di una certa complessità (come la programmazione orientata agli oggetti in accoppiata con il favoloso C++) ma si limita ad affrontare problemi di eccessiva semplicità.
(Nicola Ghelli - Reggiolo)

Purtroppo le richieste in tal senso sono piuttosto scarse; anzi: molti lettori protestano per l'*eccessiva difficoltà* di vari articoli(!).

Lo scopo delle numerose trattazioni di "basso livello" è proprio quello di invogliare i lettori più pigri ad affrontare, un po' per volta, problemi sempre più complessi, prima di passare a modalità operative davvero *toste* come quelle richieste dal nostro lettore.

Con che coraggio, poi, potremmo parlare di programmazione object oriented quando una gran quantità di lettori ha ancora problemi nel settare la stampante?

Errata corrige

Sono un insegnante che ha digitato il programma "Generatore di compiti in classe" pubblicato nel n. 83, ma non funziona.
(da alcune lettere)

Purtroppo, per un inefficiente trasferimento di da-

ti in fase di impaginazione elettronica, sono stati eliminati tutti i segni di **<** e di **>**. Per fortuna, però, è facile rintracciare i vari punti del programma in cui sono assenti.

E' sufficiente, infatti, inserirli in **tutte** le righe in cui è presente `If...istruzione` privo dei segni di maggiore e minore. Ad esempio, al posto di...

```
If X$ Chr$(13)...
```

...sostituite con...

```
If X$ <> Chr$(13)...
```

Nei punti in cui, invece, compare solo un segno di uguaglianza (ad esempio, If Stamp$ = "Printer" e simili) le righe vanno lasciate inalterate.

Lotto Amiga

Precisiamo che, per motivi tecnici, **non** è ancora disponibile la versione **Amiga** del programma **Lotto**, recensita per sistemi MS - DOS nel n. 83.

Random, vero significato

La traduzione di Ram è Random Access Memory. Ciò significa che il computer alloca a casaccio i valori nella memoria?

(Roberto R. - Roma)

No, ci mancherebbe altro! In effetti Random è sinonimo di "casuale" ma in questo caso assume un significato meno generico di quanto potrebbe sembrare.

Significa, semplicemente, che è possibile memorizzare un qualsiasi dato in una qualsiasi locazione scelta dall'utente, tra quelle disponibili, e di richiamarlo in seguito senza esser limitati da precedenti operazioni di lettura / scrittura.

E' la stessa differenza che passa tra un **file random** ed uno **sequenziale.**

Se in un file di quest'ultimo tipo sono memorizzati, ad esempio, 500 dati, per individuarne l'ultimo è indispensabile passare in rassegna i 499 dati che lo precedono.

Con un file random contenente un eguale numero di dati, invece, è possibile accedere direttamente all'ultimo (o al primo, al 345mo e così via) proprio grazie alla possibilità di indicare, a "casaccio" una qualsiasi posizione, senza tener conto di eventuali precedenti operazioni di accesso al disco.

Corsi di Computer grafica gestiti dalla Regione Lombardia

Un corso davvero interessante è quello proposto dalla Regione Lombardia al quale, ne siamo sicuri, potranno seguire altri corsi o altre iniziative in tal senso.

La durata è di **21 settimane**, per un totale di **189 ore**, trascorse "con le macchine e sulle macchine". Il corso verrà attuato nel periodo **ottobre 1991 / luglio 1992** dalla Regione Lombardia - Centro operativo regionale (Via Soderini 24 cap 20146 Milano Tel. 02/67.65.20.82; fax 02/48.91.51.84)

Il titolo di ammissione previsto *era* (la scadenza, infatti, era fissata al 15 gennaio di quest'anno) costituito dal **diploma di Scuola Media Superiore**, da produrre in copia fotostatica, o da documentata esperienza di almeno un biennio nei settori del cinema, teatro, TV, computer grafica.

Il corso era rivolto solo a **15 persone** occupate o disoccupate e la quota di iscrizione era limitata a 200 mila lire.

Speriamo che simili iniziative (la cui serietà dovrebbe esser garantita da un ente pubblico, quale appunto è la Regione Lombardia) si ripetano e che il numero dei partecipanti ammessi sia maggiore.

Gli argomenti trattati sono, ovviamente, legati alla possibilità di operare con elaboratori orientati alla gestione della grafica 2D e 3D sfruttando le nuove tecnologie che hanno modificato il modo di vedere e fare spettacolo.

Si va dalla gestione puramente grafica per mezzo di computer (creazione di fondali, mixaggio, cinemontaggi e simili) alla robotica, in grado di manovrare con estremo realismo modellini che si muovono in sottofondo *croma-key* (particolare colorazione che consente la realizzazione di trucchi, in particolare sovrapposizione di immagini).

Altre Regioni saranno abbastanza sensibili da introdurre corsi similari? Noi speriamo proprio di sì...

Amigazzetta, come procurarsela

Numerosi lettori, per distrazione(!), dimenticano di inviare, **oltre** alle **12000** lire per ciascun dischetto della serie *"Amigazzetta"*, il contributo per le spese di spedizione, come specificato sulle pagine informative, poste in fondo alla nostra rivista, circa le modalità da seguire per ricevere i nostri prodotti.

Ricordiamo, pertanto, che bisogna aggiungere, alla cifra richiesta per i prodotti editoriali, L. **3500** (imballo e spedizione per invio **semplice**) oppure L. **6000** (imballo e spedizione a mezzo **raccomandata**).

Coloro che hanno inviato una cifra minore sono quindi pregati di *integrarla* altrimenti la spedizione non può aver luogo .

Da questo mese (o, al massimo, dal prossimo), ogni problema relativo al modo di procurarsi *Amigazzetta* sarà limitato solo ai numeri arretrati della pubblicazione.

Abbiamo infatti preso la decisione di distribuire, in edicola, un prodotto di straordinario interesse per i lettori della nostra testata. Il nome della "rivista" su disco? **Computer Club Disco**, ovviamente, che vi consigliamo di non lasciarvi sfuggire.

La nuova iniziativa editoriale, infatti, sarà legata strettamente alla nostra pubblicazione e, pur se i due prodotti continueranno a vivere indipendentemente l'uno dall'altro, sarà molto più utile riferirsi contemporaneamente ad entrambi.

Per ora non vi diciamo di più. Vi invitiamo solo a ricordare il nome della nuova pubblicazione:

Computer Club Disco - Systems Editoriale

RICORDA:
Computer Club Disco - Systems Editoriale

Libro e dischetto, utente perfetto

< di Alessandro de Simone >

< Esaminare C. C. n. 86 "Manuale di confusione" editoriale C.C. n. 82 >

Sul numero scorso è stato dato libero sfogo alle perplessità che sorgono nell'affrontare la lettura - studio dei manuali di istruzione allegati alle confezioni di hardware e software.

Stavolta, invece, ci riferiamo ad un altro tipo di manualistica, parallelo al primo, che riguarda, appunto, i volumi in libera vendita presso le librerie specializzate.

A chi serve un libro

Anche qui, inutile dirlo, vi sono perplessità di vario tipo, tra le quali una, in particolare, merita la nostra attenzione.

Se si suppone che l'acquirente del libro sia già in possesso della **confezione originale** del prodotto cui si riferisce (completa, quindi, di manuale d'uso) perché mai decide di acquistarlo? Sarebbe come se, dopo aver comprato un videoregistratore, acquistassimo, a parte, un volume che ne descriva il funzionamento.

Apparentemente, dunque, ci troviamo di fronte ad un doppione, tra l'altro piuttosto costoso; le risposte che possiamo dare al successo di vendita di questa manualistica *parallela* sono diverse:

Anzitutto potrebbe essere una conferma indiretta che i manuali acclusi alle confezioni originali sono carenti, imprecisi o complessi.

In secondo luogo è probabile che l'acquirente abbia acquistato, per errore o convenienza, l'edizione originale in inglese del software cui il manuale si riferisce e decide di documentarsi su un libro italiano; pensate anche, ad esempio, alle stampanti vendute con il solo manuale in inglese (quando c'è).

In questi casi la proposta di un libro in italiano sull'uso della periferica è più che giustificato.

Un'altra ipotesi può essere quella di rifornire di un manuale personale tutti gli impiegati di un ufficio in cui si adopera un prodotto acquistato in confezione unica (e quindi dotato di un solo manuale).

L'ultima ipotesi *(a patto di non volersi nascondere dietro un dito)* riguarda tutti gli utenti di **software piratato** che, in quanto tali, non dispongono della minima documentazione.

Comunque sia, i libri posti in vendita sono numerosissimi e di qualità ormai decisamente elevata.

Tutto sta a vedere se l'utente cui sono destinati è d'accordo su tale punto di vista...

A che serve un libro

Come già detto prima, la decisione di acquistare un libro segue la necessità di documentarsi adeguatamente sul prodotto in proprio possesso.

In queste pagine ci riferiremo, per semplicità, ad un programma professionale di elevato livello (DTP **Ventura Publisher** prodotto dalla Rank Xerox) per ambiente **Ms - Dos** e cercheremo di stabilire se un utente medio può trarre vantaggio dallo ***studio*** (attenzione, escludiamo deliberatamente il termine ***lettura***) di due volumi pubblicati da altrettante case editrici, la **Mc Graw Hill** (piazza Emilia 5, cap 20129 Milano) e la **Apogeo** (Via Voghera 11/a cap 20144 Milano)

Rinviando il lettore alle due *schede* sui volumi presi in esame, ci intratterremo ora su altri aspetti generali, di sicuro interesse.

Entrambe le pubblicazioni assicurano di rivolgersi sia agli utenti **esperti** sia ai **principianti**, precisando che, in quest'ultimo caso, il principiante deve armarsi di santa pazienza e, soprattutto, seguire la lettura del volume avendo davanti il computer su cui gira Ventura.

Nonostante l'avvertenza sia ben evidenziata, riteniamo tuttavia che il principiante possa incontrare serie difficoltà

nella comprensione di manualistica relativa a programmi tanto complessi.

Purtroppo, di solito, si confonde il termine **principiante** con **autodidatta**.

Un principiante, lo dice la parola stessa, è uno che inizia, ma *non* necessariamente da *solo*.

Seguendo un corso specifico (a proposito: la casa editrice Apogeo organizza **corsi introduttivi o avanzati** per Ventura presso la sua sede o presso la sede del cliente; per informazioni tel. 02/89.40.47.22; fax: 02/89.40.45.95) seguendo un corso, dicevamo, la capacità di apprendimento, la possibilità di ottenere un immediato chiarimento dal docente, la possibilità di concentrarsi in un ambiente stimolante (immaginiamo, ovviamente, docenti responsabili e partecipanti motivati) offre l'opportunità, anche a chi è totalmente digiuno di esperienza specifica, di ottenere concreti risultati in tempi relativamente rapidissimi.

L'autodidatta, ed anche qui il termine non lascia spazio a dubbi, è di solito, invece, un utente che non ha altro aiuto se non la manualistica che può procurarsi in un modo o in un altro e articoli di riviste (oltre ad alcuni *si dice* prelevati qua e là su banche dati, amici, amici di amici e così via).

Per forza di cose, quindi, l'*autodidatta principiante solitario* è il personaggio con maggiori difficoltà che è possibile incontrare praticando un qualsiasi hobby; il bello è che il principiante autodidatta è l'animale più diffuso nello zoo informatico.

Esiste una valida manualistica per questo individuo?

Strumenti

Nel campo della didattica, in senso lato, non ha senso offrire solo uno strumento perfetto trascurando l'indispensabile efficacia di altri strumenti.

Se in una scuola, ad esempio, l'insegnamento della Geografia viene affidato ad un insegnante validissimo, questi è tuttavia destinato ad incontrare serie difficoltà nello svolgimento del suo lavoro se la stessa scuola non metterà a sua disposizione carte geografiche sufficientemente aggiornate (sui muri di una famosa scuola statale ho notato, addirittura, una carta geografica in cui erano ancora delineati i confini dell'impero fascista!).

Allo stesso modo dovrebbero far ridere i libri sulle arti marziali (o sui balli moderni) che promettono di far apprendere rapidamente i segreti delle discipline offrendo, per raggiungere lo scopo, schemi grafici di ballo o di attacco: più che la teoria è la **pratica**, in casi come questi, necessaria ed inelusibile.

Tornando a noi, un volume di informatica destinato ai **principianti** (e per di più **autodidatti**) non può prescindere dal riferimento costante ad un **dischetto**, da accludere al libro, contenente gli esempi ivi descritti.

E' vero che tale opportunità contribuirebbe a far lievitare il prezzo di copertina, ma aumenterebbe anche l'efficienza dell'apprendimento in maniera considerevole.

Inoltre, se ci è ancora concesso qualche consiglio, un volume destinato ai principianti (autodidatti o meno che siano) dovrebbe esser costituito da poche decine di pagine, allo scopo di incoraggiarne lo studio; c'è poco da fare: la prospettiva di dover studiare 500 e passa pagine tende a far venire una crisi di nervi a chiunque!

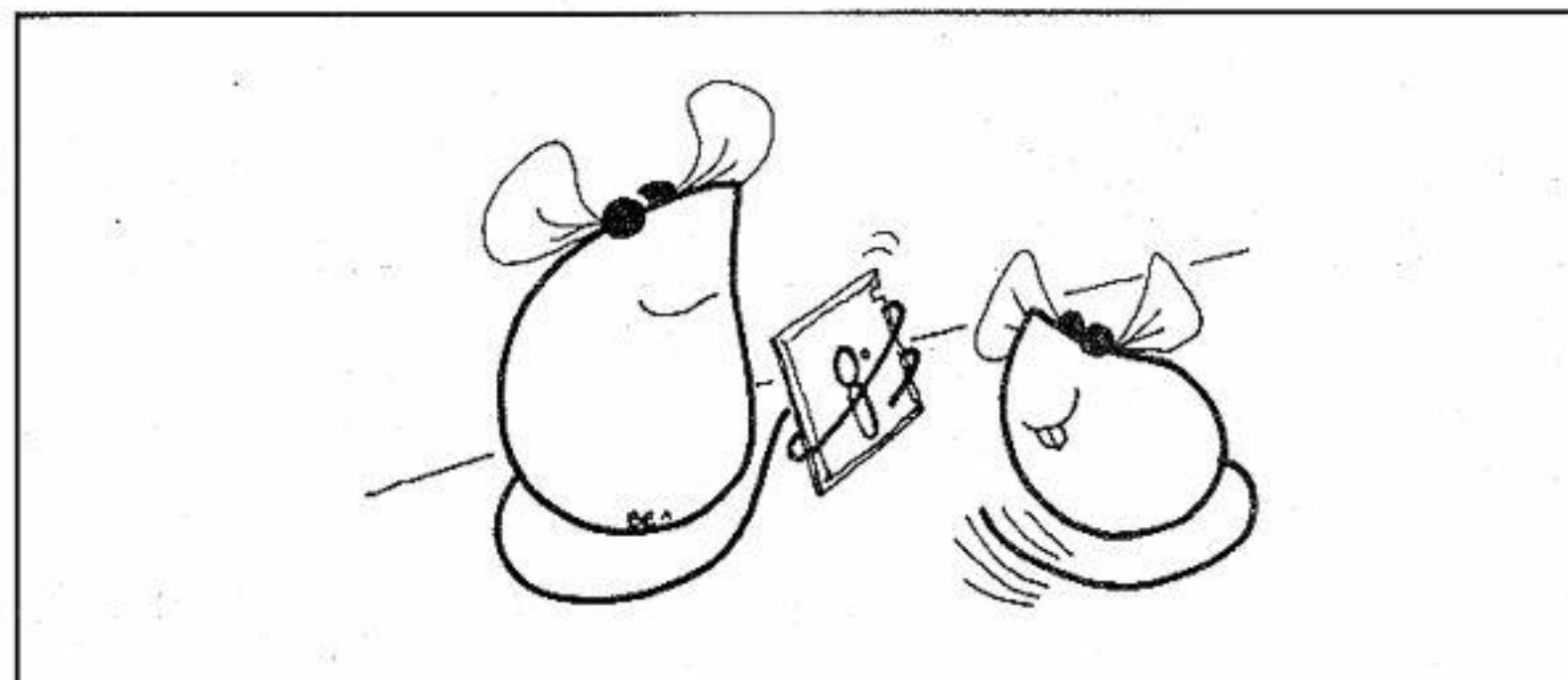
Su questo punto di vista siamo forse confortati dalla stessa casa editrice Apogeo che (parlando di altro software) propone anche volumi relativamente piccoli (valga per tutti la citazione di *Windows 3, guida al sistema operativo*, di sole **128** pagine) che potrebbero costituire una valida premessa per opere più impegnative (citiamo ancora, sempre di Apogeo: *Programmare Windows 3*, di ben **576** pagine; ne riparleremo prossimamente).

Attenzione: stiamo parlando di volumi che dovrebbero essere l'uno la premessa dell'altro; non ci riferiamo certo alla sconsiderata pubblicazione, da parte di case editrici che *non* citiamo, di veri e propri doppioni, proposti al solo scopo di riempire corposi listini e di far confusione nel potenziale lettore-acquirente.

In conclusione, e sempre secondo il nostro modesto parere, un programma professionale impegnativo, quale appunto è Ventura, dovrebbe essere affrontato in almeno due fasi, meglio se tre:

➪ Primo volume, fornito di dischetto, destinato a principianti autodidatti che siano solo in grado di lanciare il pacchetto applicativo. Il libro, in questo caso, deve insegnare esclusivamente le fasi più importanti comuni a tutti i software: come **creare** o **caricare** un file (naturalmente non complicato, ma di modesto contenuto), come **modificarlo** (senza ricorrere ad eccessivi preziosismi), come **registrarlo** (e, quindi, come gestire i file nei casi più semplici), come **stamparlo** (e, quindi, come attivare la propria stampante, evitando utilizzi sofisticati). E non pensate che un centinaio di pagine siano troppe per lo scopo prefissato; tutt'altro.

➪ Il secondo volume, anch'esso fornito di dischetto, dovrebbe passare in rassegna le opportunità più raffinate offerte dal pacchetto; 300 pagine dovrebbero esser sufficienti, considerando che il lettore, in questo caso, ha già assimilato completamente le nozioni descritte nel primo volume.

➪ Il terzo, ed ultimo(?), volume dovrebbe esser relativo ai trucchi, ai consigli ed ai collegamenti con altri software professionali. Il numero di pagine, in questo caso, può variare notevolmente; i dischetti, invece, dovrebbero essere almeno un paio, in grado cioè di contenere i numerosi ed inevitabilmente lunghi file di esempio citati nel volume. ✍

Ventura 2, secondo Mc Graw Hill

Ventura 2, guida completa
di M. Holt e R. Birmele
Mc Graw Hill
L. 65.000

Il volume consta di **500** pagine ed, effettivamente, passa in rassegna tutte le opportunità offerte dal potente DTP della Rank Xerox.

E' suddiviso in **quattro** parti (per un totale di 14 capitoli), oltre a **tre** appendici ed all'indice analitico.

Nella prima parte, ovviamente, ci si preoccupa di descrivere per grandi linee che cos'è un **DTP** (**D**esk **T**op **P**ublishing = programma di impaginazione), si descrivono i vari termini usati in *tipografia* ed in sede di *composizione* e di *impaginazione*. Si passano in rassegna le parti che possono far parte di una pagina-tipo e si stabilisce il significato di *foglio di stile*. Si accenna alle possibilità grafiche e all'opportunità di ottenere testi ruotati. La differente forma del documento su video e su carta, oltre alla descrizione del metodo di stampa, conclude il quarto capitolo; la memorizzazione dei file, la loro ubicazione ed il corrispondente trattamento vengono affrontati dall'origine (parlando dei comandi di Ms - Dos) fino alla manipolazione specifica in ambiente Ventura. La seconda parte inizia con la personalizzazione della *tastiera*; prosegue con alcuni esempi sulla creazione di una pagina di testo sia in casi semplici (volantini, newsletter), sia in casi più complessi (creazione di indici automatici); si passa quindi alla manipolazione di *strumenti grafici* (retini, file grafici) ed alla fusione di una pluralità di capitoli in un'unica *pubblicazione*; precisazioni sull'opzione multi-capitolo concludono la seconda parte.

La connessione con diverse *stampanti*, l'uso di vari word processor (da cui importare file in Ventura), l'attribuzione di caratteristiche speciali al testo, l'editing di complesse *formule matematiche*, la gestione di variabili e di riferimenti incrociati sono gli argomenti trattati nei tre capitoli della terza parte.

La quarta parte è costituita da un unico capitolo (di circa 60 pagine) che rappresenta, in effetti, un dettagliatissimo *compendio alfabetico* di tutti gli argomenti trattati nel volume; si tratta di un vero e proprio mini manuale di pronto intervento, utilissimo per coloro che, avendo ben appreso l'impostazione generale di Ventura, vogliono rintracciare, con la massima rapidità, alcuni dettagli poco noti: una specie, insomma, di help in linea... cartaceo!

Delle appendici, la prima è forse troppo limitata dal momento che, in sole due paginette, pretende di fornire indicazioni utili per l'installazione di Ventura(!); la seconda, invece, è un utile e completo elenco dei codici di attributo del carattere, dei simboli delle equazioni, dei caratteri greci e dei set di caratteri da inserire eventualmente in fase di digitazione da w/p, dei nomi e dei numeri delle font usate da Ventura; la terza appendice, che precede l'indice analitico, elenca sia la successione dei tasti da premere, in ambiente Ventura, per far apparire *caratteri speciali* come il copyright ©, il marchio registrato ® ed altri; sia i tasti da premere per attivare, rapidamente, alcune funzioni speciali di Ventura.

Impressioni d'uso

Il volume, come già detto, rappresenta una guida completa all'uso di Ventura; anzi, spesso si intrattiene in precisazioni sulle differenze riscontrabili tra le precedenti versioni (V. 1, V 1.1) e quelle più attuali (V. 2, professionale).

Il pacchetto Rank Xerox preso in esame dagli autori è quello relativo alla **versione italiana** e tutte le illustrazioni e gli esempi d'uso pubblicati si riferiscono a tale versione.

L'impostazione del libro è di tipo tradizionale: numerose illustrazioni, a corredo del testo, contribuiscono a meglio chiarire gli argomenti affrontati; questi sono ordinatamente suddivisi nelle varie parti e capitoli, e di rado si trovano informazioni, relative ad uno stesso argomento, "disseminate" in altre parti del volume:

tutto è concentrato ed esaurientemente spiegato in specifiche sezioni.

Abbiamo però notato una certa ripetitività, relativa ad informazioni che potevano esser date, all'inizio, una volta per tutte.

Lo stile di esposizione ricorre, a volte, a periodi piuttosto lunghi, costringendo ad una rilettura degli stessi per una migliore assimilazione.

Sono pochi, a nostro parere, gli esempi pratici che descrivano dettagliatamente, e con completezza, tutte le azioni da compiere per realizzare una certa operazione.

Alcuni errori di battitura, pur se raramente riscontrati, fanno intuire che, forse per la fretta di andare in stampa, il volume non è stato riesaminato attivando la correzione ortografica. Il libro sarà apprezzato da chi è abituato alla manualistica tradizionale in cui, cioè, l'ordinata successione di argomenti, suddivisi secondo una classificazione poco frazionata, consente di apprenderli in maniera sistematica, esauriente e completa.

Ventura 2, secondo Apogeo

Ventura 2,
estensione professionale
di T.Nace e D. W. Harris
Apogeo
L. 68000

Il volume consta di circa **500** pagine, introduzione ed indici compresi, garantendo (c'è scritto sulla copertina) la trattazione di *consigli, trucchi & miracoli.*

Anche in questo caso le promesse della copertina vengono mantenute in **sei** sezioni (per un totale di ben **31** capitoli) ed un'appendice suddivisa in due parti.

La prima sezione segue la tradizione di introdurre il lettore alle possibilità offerte dal potente pacchetto della Rank Xerox ed approfitta per illustrare le differenze tra Ventura 1.1, Ventura 2 e l'estensione professionale (sempre per la **versione italiana**).

La seconda sezione è relativa all'hardware da usare con Ventura, soffermandosi su consigli pratici per evitare investimenti inadeguati sia per ciò che riguarda *computer*, sia *monitor* sia *stampanti*, sia *scanner*.

La gestione dei file occupa l'intera terza sezione, mentre la quarta, relativa alla grafica, si occupa dei principali *pacchetti applicativi* che è possibile usare parallelamente con Ventura.

Le *fonti* dei caratteri occupano l'intera quinta sezione che descrive anche i metodi per caricare ed installare nuovi font sia per stampanti Postscript che tradizionali.

La sesta ed ultima parte, dall'accattivante titolo *Specialità*, è un'opportuna miscellanea di trucchi e consigli su come comportarsi in casi particolari e su come superare piccoli problemi che dovessero presentarsi.

La prima appendice è un manuale di istruzioni su come realizzare, passo per passo, una newsletter, partendo dal word processor, passando attraverso la creazione di frame, l'uso corretto di tabulazioni, l'impostazione di marcatori, piè di pagina, strutture grafiche e così via.

La seconda appendice è un utilissimo *mini vocabolario* relativo alla traduzione di *tutti* i comandi presenti nell'edizione in inglese. Questa appendice sarà preziosissima per coloro che, non conoscendo l'inglese, possono avere difficoltà nell'uso del pacchetto Ventura in edizione anglofona.

Un completo indice analitico di 21 pagine completa il volume.

Impressioni d'uso

Il volume si presenta ricchissimo di illustrazioni e tabelle ed è impaginato in uno stile grafico moderno e di sicuro effetto; insomma, un classico manuale *all'americana*, in cui un apparente disordine nasconde, in realtà, un collegamento preciso ed esauriente tra vari capitoli e sezioni.

Tra i punti certamente positivi possiamo citare la felice forma di *domande / risposte*, presente alla fine di tutti i capitoli, in grado di meglio evidenziare eventuali problemi che dovessero insorgere nell'uso del pacchetto.

Semplicemente geniale l'idea di pubblicare, con sufficiente completezza, interi paragrafi su informazioni tecniche relative a stampanti, monitor e software per detti. Descrizioni, raramente superficiali, più spesso approfondite, su altri *programmi* da usare con Ventura (e su quelli presenti nel dischetto utilità di Ventura stesso), saranno particolarmente apprezzate da chi intende sfruttare fino in fondo le capacità di importare file grafici e di testo da altri pacchetti software disponibili sul mercato.

Per contro, il volume della Apogeo presuppone una notevole attenzione da parte del lettore, che potrebbe sentirsi frastornato da alcuni salti apparentemente irrazionali da un argomento all'altro; tale forma di esposizione, al contrario, rende meglio l'idea che forse Ventura, come ogni altro pacchetto complesso, non può essere affrontato per compartimenti stagni, ma è necessario accennare ad una certa potenzialità mentre se ne descrive un'altra che, a sua volta, tira in ballo un nuovo problema... e così via.

I *consigli* di cui è disseminato il manuale sono tutti ottimi, pertinenti ed efficacemente descritti. La stessa impaginazione degli indici analitici ed alfabetici, completi ed esaurienti, fanno capire che nella stesura del volume si è cercato di rendere quanto più semplice possibile la consultazione ed il rintracciamento dei numerosissimi argomenti.

Insomma, è un libro da leggere *due* volte: la prima volta tutta di un fiato (saltando, magari, le operazioni più complesse e meno frequenti). La seconda volta per diventare realmente padroni di Ventura, sfruttando fino in fondo i preziosi consigli riportati nel volume.

Conclusioni

Grosso modo, i due libri trattano esaurientemente tutti gli argomenti relativi ad un uso **professionale** e **completo** di Ventura. Il testo dell'Apogeo, in un numero praticamente eguale di pagine, riesce tuttavia ad inserire un maggior numero di figure e tabelle, oltre ad utili informazioni su programmi commerciali di sicuro interesse per chi lavora molto con file grafici.

Entrambi i volumi vengono forniti, purtroppo, privi di un dischetto che a nostro parere, come già detto, è ormai indispensabile per l'ottimale comprensione di un pacchetto professionale.

I due libri, comunque, sono ugualmente validi ed indicano senza ombra di dubbio la notevole qualità della manualistica oggi disponibile nelle librerie specializzate.

Come ti acchiappo il virus

< di Alessandro de Simone >

< Esaminare editoriale C. C. n. 84>

La redazione di un giornale di informatica, come ogni altro ufficio in cui circolano dischetti di provenienza disparata, è un terreno fertile per la diffusione di virus inclusi, volontariamente o meno, nei dischetti che capita di provare.

Inutile dire che, da qualche anno a questa parte, sono state prese opportune precauzioni onde evitare contagi: in parole povere, ogni volta che in redazione "entra" un dischetto (qualunque sia la sua provenienza) viene sottoposto ad un test per verificare la presenza di virus. Allo scopo è stato installato un banale **file batch** che esplora l'intera superficie del dischetto.

Sia per **Ms - Dos**, che per **Amiga**, non sono mancate le sorprese soprattutto da coloro che, assolutamente insospettabili come *untori*, portavano inconsapevolmente con sè virus di vario tipo.

Come si diffonde un virus

La casistica Amiga è, purtroppo, molto più varia che nel caso Ms - Dos. L'aggiornamento dei programmi antivirus viene effettuato non appena il nostro *Domenico Pavone*, viaggiando quotidianamente per telefono nelle BBS di tutto il mondo (o quasi...) viene a conoscenza di qualche nuovo antidoto.

Con Amiga, infatti, non appena viene inserito un dischetto in un drive, il sistema operativo provvede ad attivare un programma inserito nel bootblock del dischetto stesso.

Tale accorgimento, in origine, era stato adottato dai progettisti Commodore per facilitare al massimo l'interfaccia utente; in questo modo, infatti, non era necessario comunicare al sistema l'avvenuto inserimento di un nuovo disco, ma automaticamente venivano catturate tutte le informazioni necessarie per la successiva manipolazione dei file.

Poiché l'attivazione della procedura era totalmente automatica (ed altrettanto trasparente per l'utente finale) a qualche buontempone venne l'idea di modificare il programma di auto-identificazione in modo da fargli contenere anche un virus; vale a dire che, tutte le volte che si inseriva quel dischetto, veniva attivata una procedura destinata a compiere danni più o meno rilevanti non solo sul dischetto in questione (o nella stessa RAM di Amiga), ma in tutti i successivi dischetti che si fossero inseriti in qualsiasi drive.

Con i sistemi Ms - Dos, invece, **non** è sufficiente inserire un dischetto contenente un virus; per attivarlo è infatti necessario che quest'ultimo sia contenuto, naturalmente in modo subdolo, all'interno di un programma da lanciare. Se tale programma non viene lanciato il virus resterà innocuo.

Con i sistemi Ms - Dos, pertanto, l'unico modo di lanciare un virus senza che l'utente se ne accorga è quello di inserirlo nel boot block del dischetto di sistema.

Se, però, si ha l'accortezza di lanciare il proprio sistema servendosi sempre dello stesso dischetto di cui si conosce perfettamente il contenuto, il pericolo di infezioni diminuisce notevolmente.

Gli antivirus

Non staremo, qui, a dire come funziona un programma antivirus.

Ci limiteremo a ricordare che è un particolare programma in grado di intercettare *spezzoni* di programmi sospetti e di avvertire l'utente sul pericolo che si corre.

I programmi antivirus devono **continuamente** essere aggiornati dal momento che per **ogni** antivirus scoperto c'è **almeno** un "terrorista informatico" in grado di annichilirne le difese.

Per risolvere il problema del continuo aggiornamento vi sono sostanzialmente tre metodi.

➪ Il primo, più banale (e scomodo) consiste nel divulgare un antivirus per ciascun virus sconosciuto. Questo metodo poteva andar bene fino a qualche anno fa, quando i virus diffusi erano poche decine. Oggi sarebbe troppo scomodo verificare un dischetto per mezzo di alcune centinaia di programmi specifici!

➪ Il secondo consiste nel proporre, periodicamente, una nuova versione di uno stesso programma antivirus, che in questo caso contiene, al suo interno, tutte le informazioni utili per intercettare i virus conosciuti fino al momento del suo rilascio.

E' il caso del ben noto **ScanXX** (per sistemi **Ms - Dos**) che, dalle prime versioni diffuse qualche anno addietro, è giunto fino alla versione **Scan77** (parliamo del mese di maggio 91; oggi forse si è già oltre la 77-ma versione).

➪ Il terzo metodo seguito per aggiornare la lista dei virus è quello di proporre un programma strutturato in modo da ricevere gli aggiornamenti sotto forma di file da aggiungere nella directory del programma stesso.

E' il caso di **Norton Anti Virus (NAV)**, che offre un servizio gratuito di post vendita mediante due banche dati, la prima negli **USA** e la seconda in **Inghilterra**.

Da queste due **BBS** (il cui numero di telefono è riportato sul manuale della confezione originale) chiunque può prelevare i file aggiornati, che hanno tutti suffisso **.DEF** e sono riconoscibili dallo stesso nome (Update01.def, update02.def, eccetera).

Inutile dire che tali file, da soli, non servono a nulla se non si possiede il programma NAV.

Dopo aver inserito nella directory di NAV il file prelevato dalla BBS, il programma sarà in grado di individuare qualsiasi virus tra quelli contenuti in tutti i file di suffisso .DEF che, un po' per volta, si è costretti a "collezionare".

Il motivo di un tale sistema di aggiornamento è legato alla necessità di offrire un servizio post-vendita veloce, gratuito (per chi ha acquistato il programma originale) e relativamente economico come spese di esercizio.

I file .DEF sono, infatti, disponibili sulle BBS in formato compatto; all'origine occupano poche decine di migliaia di byte con una conseguente necessità di collegamento, via modem, piuttosto ridotta.

Ma funziona?

Un programma antivirus è uno di quei programmi di cui non si può sapere se effettivamente funzionano.

E' un po' come la garanzia di un rifugio antiatomico: se non funziona correttamente (in seguito ad un conflitto nucleare) a chi rivolgersi per far valere i propri diritti?

In altre parole bisognerebbe avere a disposizione dischetti infettati per vedere l'efficacia del programma.

Del resto, chi è tanto imprudente da inserire un dischetto infettato (in un sistema, magari, dotato di hard disk di 200 megabyte...) per verificare l'efficienza di un antivirus?

Con NAV è tuttavia possibile verificarne l'efficienza limitandosi a compiere alcune operazioni del tutto innocue su un qualsiasi dischetto (anche hard disk).

NAV, per chi non lo sapesse, si installa in un computer Ms - Dos in modo tale che, all'accensione, viene attivato un particolare intercettore in grado di evidenziare anomalie sospette durante il funzionamento di qualsiasi programma si faccia girare da quel momento in poi.

Con un sistema Ms - Dos, infatti, un virus può insediarsi, e proliferare, solo se riesce ad inserirsi in programmi con estensione **.EXE** oppure **.COM** in modo tale che l'ignaro utente, lanciando il programma (che, appunto, può avere solo estensione .EXE o .COM) innesca anche il contagio.

Per compiere tale operazione, tuttavia, il virus è costretto, inserendosi nel programma da infettare, ad allungarne la dimensione, anche se di pochi byte.

Se un programma non ancora infettato occupa uno spazio di 10000 byte, dopo l'infezione, ad esempio, ne potrà occupare 10114.

NAV, grazie al fatto che è installato permanentemente nella memoria del computer, può individuare qualsiasi tentativo di modifica di file .EXE e .COM. In casi come questo, interrompe qualsiasi operazione in corso e visualizza una finestra di allarme in modo che l'utente possa prendere gli opportuni provvedimenti.

Un virus innocuo

In pratica NAV verifica se una qualsiasi operazione di manipolazione dei file tende a modificare la lunghezza di tutti i file .COM ed .EXE precedentemente indicati dall'utente (e memorizzati in file lunghi ciascuno 77 byte e dotati di estensione ._XE oppure ._OM).

E' però possibile ingannare NAV, e tutti i programmi che funzionano in modo analogo, facendo credere che un "qualcosa" stia agendo in modo inconsueto. Allo scopo, seguite tutte le indicazioni qui riportate:

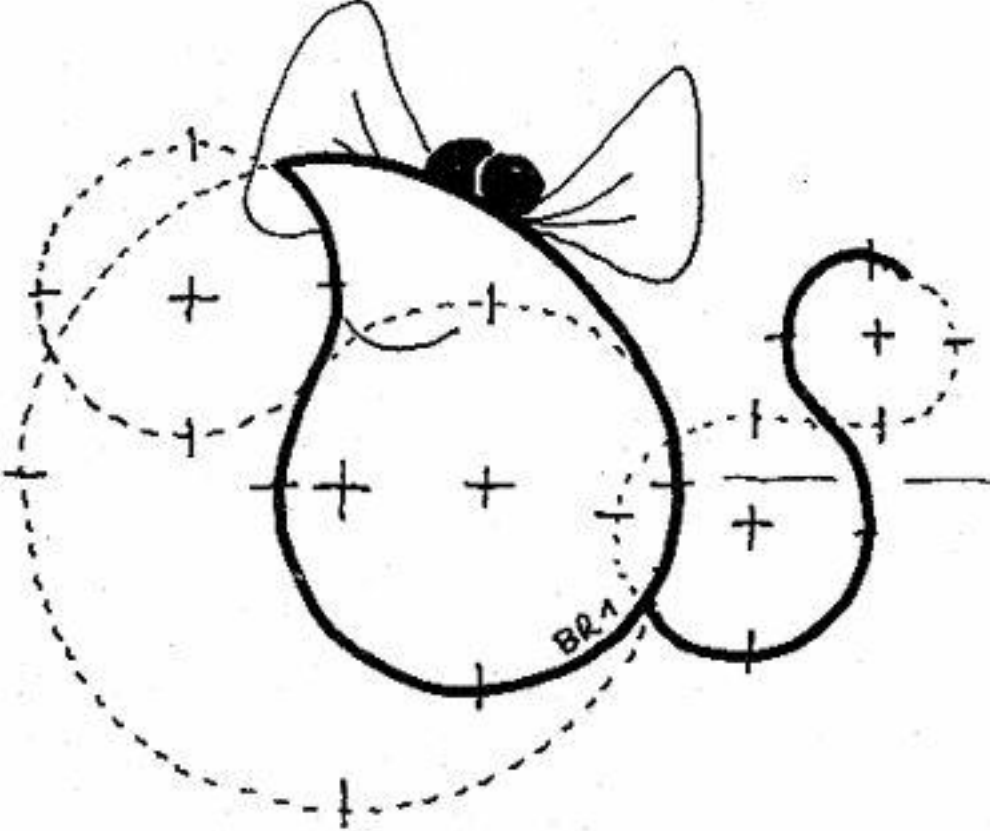
Create, su un dischetto posto nel drive **a:**, un file .EXE utilizzando il comando copy con:

```
a> copy con: prova.exe
Questa e' una prova.
^Z
```

L'ultimo rigo, come è noto, si ottiene premendo Control + Z ed il tasto return. Subito compare il messaggio: *1 file copiato* (o simile).

Allo stesso modo, create un altro file, ad esempio Ciao:

```
A> copy con:ciao
questo e' un altro file di
prova
^Z
```

Ora caricate NAV ed attivate la modalità **inoculate**, in modo tale, cioè, che il file `prova.exe` (ora presente nel drive a:) venga inserito nella lista dei file da controllare continuamente in tempo reale. Usciti da NAV, se avete un'utility tipo PC Tools potrete notare che, nel dischetto a, è ora presente anche il file `prova._xe` (con un semplice comando Dir non è possibile verificarne la presenza perché è un file nascosto).

A questo punto provate ad eseguire:

```
A> copy ciao + ciao prova.exe
```

...nel tentativo di alterare il contenuto del file .exe.

E' probabile che la prima volta non succeda nulla. Se, però, eseguite nuovamente lo stesso comando, apparirà una finestra rossa, accompagnata da un sonoro segnale di allarme, confermando che l'operazione di alterazione di un file eseguibile non è passata inosservata.

Il lettore, per proprio conto, sarà certamente in grado di stabilire da solo altri trucchi per attivare programmi antivirus che agiscono considerando modifiche apportate a file o a particolari settori di un dischetto. ✍

Pitagora, il suo teorema e gli sfidanti

< di Gianni Gottardo >

< Esaminare C. C. n. 85 >

I programmi di queste pagine sono stati selezionati tra quelli inviati dai lettori "sfidanti" che hanno risposto alla sfida lanciata nel numero **85** in cui si chiedeva di **dimostrare il teorema di Pitagora sui triangoli rettangoli**.

Il primo programma pubblicato è in semplice **AmigaBasic**; gli altri due in Amos; tutti e tre rispondono pienamente alle richieste esposte nella sfida.

AmigaBasic

Non appena si lancia, compare un triangolo di partenza, **rettangolo** e **isoscele**, completo di quadrati costruiti sui cateti e che può raggiungere le dimensioni minime di un pixel per cateto.

Mediante il mouse è possibile modificare le dimensioni dei due cateti; durante la modifica le quattro figure geometriche si adeguano alla nuova impostazione fornendo i dati relativi alla lunghezza di ogni lato e di ogni area.

Se, invece, le misure sono inadeguate, il software provvede a spostare il punto in cui si trova il vertice in modo che tutto rientri nell'area del video; ovviamente però, questa è limitata, e la figura non può essere ampliata più di tanto.

Come già detto, è possibile modificare le dimensioni del triangolo mediante quattro caselle situate in alto a destra, contraddistinte da altrettante iniziali **N, E, S, W**, dei quattro punti cardinali.

In realtà i programmi inviati dal nostro lettore sono due, ma la pubblicazione è relativa solo al primo sia perché è più breve, sia perché gira meglio anche sui sistemi dotati di soli 512 K Ram, sia perché le differenze tra i due non sono notevoli.

Il programma forse non vanta di originalità dal momento che sono stati semplicemente "eseguiti gli ordini" della sfida; nonostante ciò il lavoro è degno di nota e, soprattutto, svolge il suo compito egregiamente.

Amos

Il secondo ed il terzo listato sono, come già detto, scritti in Amos. Quello di **Enrico Marchesin** è il più veloce e consente di modificare le dimensioni del triangolo cliccano sui vertici e "trascinandoli" lungo lo schermo.

Non riportiamo la descrizione del suo funzionamento perché l'autore, sbadatamente, ha dimenticato di inserire nella busta il disegno cui faceva riferimento nell'articolo.

Abbiamo provato a ricostruire la figura, ma evidentemente c'era qualcosa che non andava per il verso giusto dal momento che alcune informazioni erano in contraddizione tra loro ed il testo era stato scritto con chissà quale word processor (venivano visualizzati, in certi punti, diversi caratteri strani).

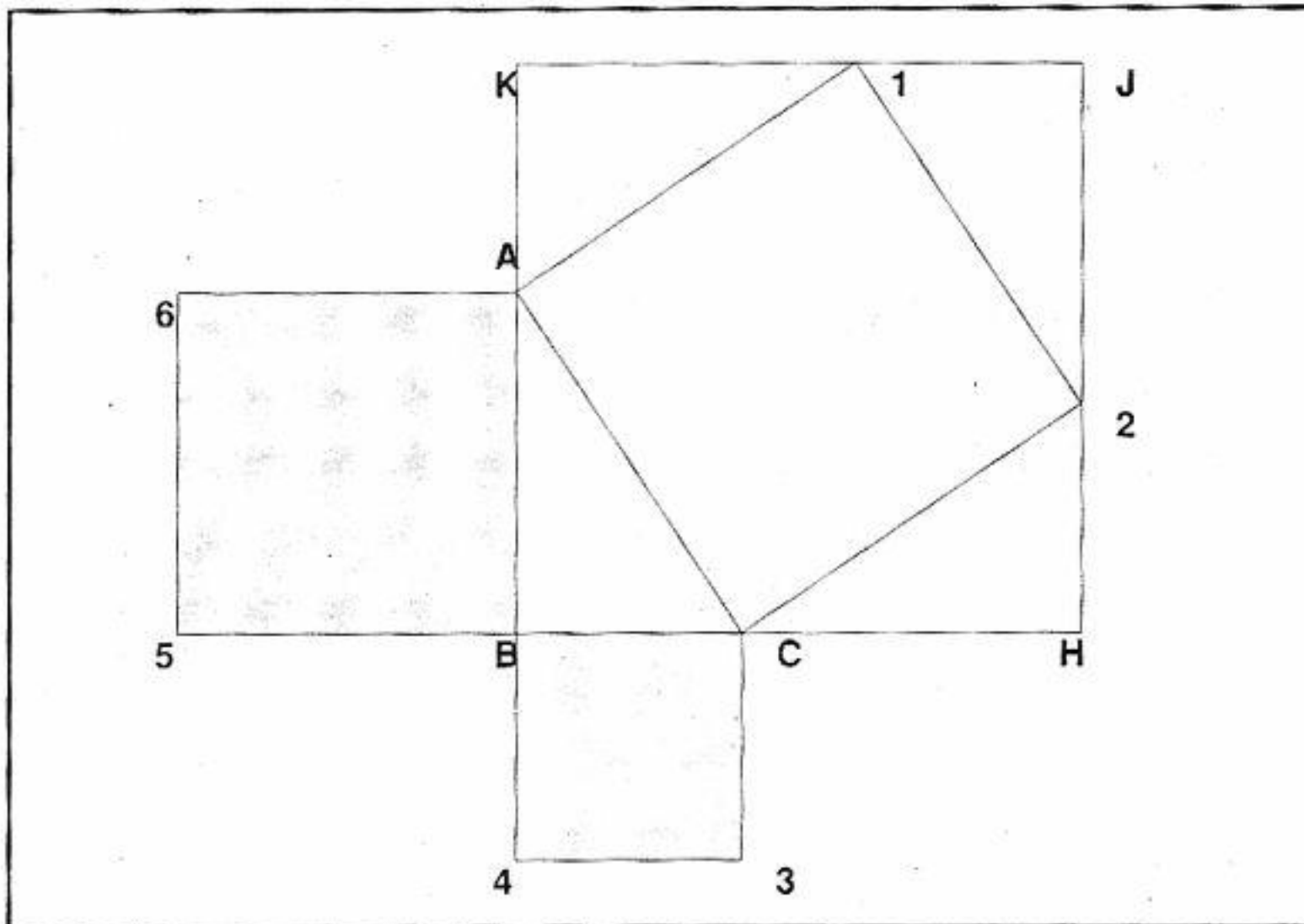
Morale della favola: siate più precisi e, soprattutto, telefonate **prima** di inviare il materiale, per evitare spiacevoli sorprese.

Anche il **secondo listato** in Amos, di **Simone Riccetti** di Crema, usa il mouse per modificare la figura.

La prima sezione del listato, di inizializzazione, si preoccupa di aprire e cancellare uno schermo in alta risoluzione PAL a 4 colori ed a nascondere il cursore. Viene quindi cambiato l'aspetto del puntatore del mouse e si provvede a delimitate le zone di schermo "sensibili" alla sua selezione.

Per quanto riguarda le variabili usate è bene far riferimento alla figura.

Ogni vertice dei poligoni (triangolo e tre quadrati) componenti il disegno, ha coor-

```
'----Teorema di Pitagora in  AMIGABASIC----
'--Dimostrazione matematica e geometrica---
'--------Scritto da Gottardo Gianni--------
bx=100:by=50:bas=70:alt=35

coord:
Ax=bx:Ay=by+alt:cy=by:cx=bx+bas:dy=Ay:Dx=bx-(alt*2):ex=cx:Ey=by-(bas/2)
Hy=Ay+(bas/2):Hx=Ax+(alt*2):iy=cy+(bas/2):Ix=cx+(alt*2)
IF Hy>180 AND Ix>580 THEN bas=bas-2:alt=alt-1:GOTO coord
IF Hy>180 AND Ey<2 THEN bas=bas-2:GOTO coord'
IF Ix>580 AND Dx<4 THEN bas=bas+2:GOTO coord
IF Hy>180 THEN by=by-1:GOTO coord
IF Ix>580 THEN bx=bx-1:GOTO coord
IF Ey<1 THEN by=by+1:GOTO coord
IF Dx<2 THEN bx=bx+2:GOTO coord
IF alt=0 THEN alt=alt+1:GOTO coord
IF bas=0 THEN bas=bas+2:GOTO coord

Disegno:
CLS
LINE (bx,by)-(Dx,dy),,b:LINE (bx,by)-(ex,Ey),,b:LINE (Ax,Ay)-(Hx,Hy)
LINE (Hx,Hy)-(Ix,iy):LINE (Ix,iy)-(cx,cy):LINE (cx,cy)-(Ax,Ay)
LOCATE 2,74:PRINT "N"
LOCATE 3,72:PRINT "W   E"
LOCATE 4,74:PRINT "S"
LINE (581,8)-(593,32),,b:LINE (565,16)-(610,24),,b
LOCATE 16,60:PRINT "1 Cateto = ";alt
LOCATE 17,60:PRINT "2 Cateto = ";bas/2
ipot=(alt*alt)+(bas/2*(bas/2))
LOCATE 18,60:PRINT "Ipotenusa = ";(ipot)^(.5)
LOCATE 19,60:PRINT "  1 Area = ";alt*alt
LOCATE 20,60:PRINT "  2 Area = ";bas/2*(bas/2)
LOCATE 21,60:PRINT "  3 Atea = ";alt*alt+bas/2*(bas/2)
GOTO ContrMouse

ContrMouse:
IF MOUSE(0)=0 THEN ContrMouse
IF ABS(X-MOUSE(1))>2 THEN GOTO Camb
IF ABS(y-MOUSE(2))>3 THEN ContrMouse

Camb:
Mx=MOUSE(1):My=MOUSE(2)
IF Mx>581 AND Mx<593 THEN GOTO VarAlt
IF My>16 AND My<24 THEN GOTO VarBas
IF INKEY$="d" THEN GOTO Dati
GOTO ContrMouse

VarAlt:
IF My>8 AND My<16 THEN alt=alt-1:GOTO coord
IF My>24 AND My<32 THEN alt=alt+1:GOTO coord
GOTO ContrMouse

VarBas:
IF Mx>565 AND Mx<581 THEN bas=bas-2:GOTO coord
IF Mx>593 AND Mx<610 THEN bas=bas+2:GOTO coord
GOTO ContrMouse
```

La figura cui fare riferimento per il programma di Simone Riccetti

dinate **X** e **Y** sullo schermo contenute in alcune variabili e aggiornate di continuo. Per i punti **A**, **B**, **1**, ad esempio, le coppie di coordinate sono, rispettivamente, **AX** e **AY**; **BX** e **BY**; **X1** e **Y1**, eccetera.

Il programma, quindi, posiziona il punto **B** nello schermo, fissando le sue coordinate iniziali. Altre variabili necessarie al tracciamento del disegno sono le lunghezze dei lati **AB**, **BC** e **AC**.

Inizialmente la lunghezza dei cateti viene impostata a **15 pixel**.

I punti **K**, **J**, **H** hanno la loro importanza. Nel programma, infatti, è necessario calcolare le coordinate dei punti **1** e **2** in modo che i segmenti **A1** e **C2** siano perpendicolari ad **AC**. Per fare ciò basta ricorrere ad uno stratagemma: prolungando **BC**, e tracciando la sua perpendicolare passante per il punto **2**, troviamo il punto **H**. Notiamo, allora, che CH è uguale a AB e che H2 è uguale a BC. E' facile dedurre che la coordinata X del punto 2 è uguale a quella del punto C più la distanza compresa tra C e H.

Si deducono le formule relative alle coordinate orizzontali e verticali dei punti 1 e 2...

X1 = AX + AB
Y1 = AY - BC
X2 = CX + AB
Y2 = CY - BC

...che consentono di tracciare il quadrato costruito su AC.

Noterete che, nel programma, i valori di AB e BC sono stati **raddoppiati** quando vengono usati per calcolare una coordinata sull'asse X. Questo perchè l'alta risoluzione è tale che in orizzontale i pixel hanno una risoluzione maggiore che in verticale; senza un opportuno correttivo la figura sarebbe risultata schiacciata.

La sezione di calcolo si occupa di aggiornare le coordinate dei vertici dei poligoni in base alle modifiche apportate alla lunghezza dei cateti. Notate il carattere di cancelletto (#) che segue la variabile contenente la misura dell'ipotenusa (**AC**). E' stato, infatti, necessario ricorrervi per avere una precisione che tenesse conto anche dei decimali dei numeri, pena risultati non precisi, ma approssimati.

Il programma provvede, quindi, a controllare che la figura (e in particolar modo il quadrato costruito sull'ipotenusa) non esca dai limiti dello schermo e, nel caso in cui ciò avvenga, cerca di avere l'immagine interamente visibile. Le variabili FLAGX e FLAGY servono a impedire che il computer centri un'immagine che è appena stata spostata per i motivi prima citati.

La sezione di disegno setta un colore visibile e richiama la subroutine TRACCIA, che serve a tracciare le linee che compongono il disegno ed a visualizzare le lettere dei vertici del triangolo. Vengono poi visualizzati i dati relativi alle aree dei tre quadrati e alla lunghezza dei lati del triangolo.

Infine vengono disegnati i **pulsanti** per modificare la figura (e che corrispondono alle zone sensibili precedentemente preparate).

Per creare la freccia in alto è stata usata la lettera **"A"**, alla quale è stata cancellata la barra orizzontale, trasformandola in una **"V"** capovolta.

Il programma prosegue attendendo un comando. Quando viene premuto il pulsante sinistro del mouse, si controlla in quale zona l'utente ha cliccato. Se non è stata selezionata nessuna area predefinita, si attende una nuova pressione del tasto sinistro, altrimenti vengono modificate le misure dei cateti in base al pulsante selezionato.

Il programma impedisce che i lati del triangolo possano essere di misura inferiore all'unità.

Se si seleziona il pulsante contenente la lettera **"E"** (posta al centro delle frecce), si esce dal programma.

La freccia a destra allunga il cateto BC, mentre quella a sinistra lo accorcia; quella in alto allunga il cateto AB, mentre quella in basso lo accorcia. Tenendo premuto il pulsante l'immagine verrà modificata più velocemente.

Prima di calcolare le nuove posizioni dei vertici viene ancora richiamata la subroutine TRACCIA dopo aver selezionato il colore di sfondo, in modo da cancellare la figura e le lettere, senza dover cancellare lo schermo intero (soluzione che farebbe lampeggiare fastidiosamente il video e ridurrebbe leggermente la velocità di risposta del programma).

Il listato è soggetto a numerosi miglioramenti. Si potrebbero, ad esempio, caricare **sprites** per i pulsanti (di sicuro più graziosi di quelli stilizzati che genera il programma).

Oppure si potrebbe dare la possibilità all'alunno di inserire direttamente da tastiera valori per i lati, affinchè il computer calcoli quello mancante, permettendogli di risolvere piccoli problemi. Si potrebbe anche far calcolare al computer una scala di rappresentazione che consenta di avere sempre sott'occhio l'intera figura, anche se le misure dei lati risultassero notevoli.

Gli altri

Altri lettori hanno risposto alla sfida. Da segnalare, in particolare, **Andrea Naimoli** di Nuoro che ha inviato un listato in **Amos** simile a quello di *Riccetti*, ben curato nella forma, ma affetto da un fastidioso sfarfallio che si manifesta durante la modifica della figura.

Matteo Macoratti (di Tolmezzo - UD) a causa di alcuni preziosismi grafici (ma chi te lo ha fatto fare? Avevamo precisato che i listati delle sfide devono essere brevissimi...) ha appesantito il programma e la velocità di elaborazione, ovviamente, ne risente. Se non bastasse, inoltre, l'eccessivo allungamento di uno dei cateti induce il programma in errore e, benché questo venga "intercettato", il programma riparte dall'inizio.

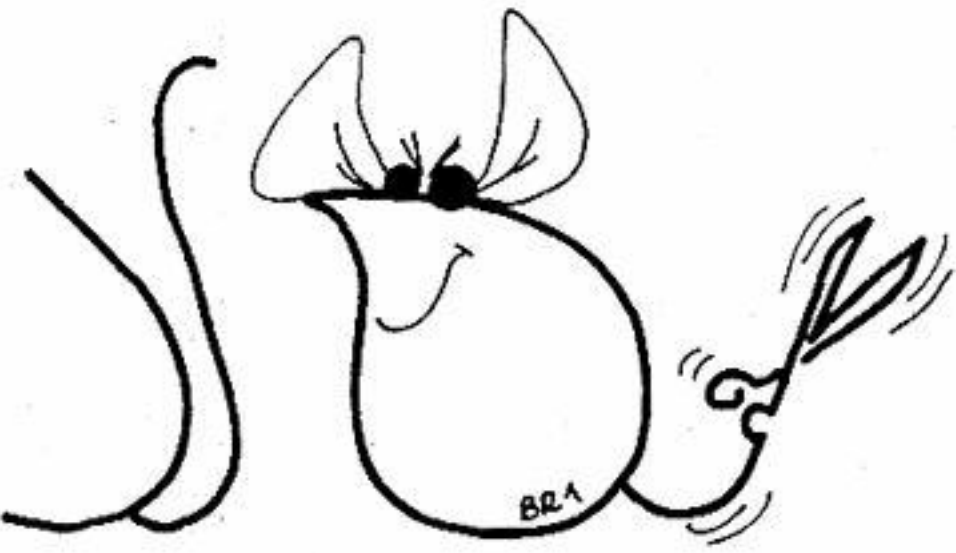
Il suo programma, tuttavia, è molto curato nei dettagli e, se fatto "girare" in forma compilata, merita un certo plauso, soprattutto se sarà possibile individuare la pressione continua del mouse.

Federico Lo Monaco non ha invece provveduto ad intercettare l'errore dovuto ad un sovradimensionamento dei cateti, ed il programma, benché molto bello (in alta risoluzione interlacciata) genera un errore con conseguente interruzione.

Davide Piscitelli (di S. felice Circeo - Ce), forse a causa della fretta, ha invece inviato un programma probabilmente incompleto, dal momento che l'incremento dei cateti avviene contemporaneamente (il triangolo rettangolo, insomma, rimane sempre isoscele).

Agli autori dei programmi pubblicati, purtroppo, non può essere consegnato il favoloso premio promesso (il funzionale appendiabiti di stile post-moderno) a causa del deterioramento delle nostre scorte: tutti i **chiodi** custoditi in cantina, infatti, si sono arrugginiti.

In alternativa abbiamo deciso di inviare, come segno di tangibile compenso, un pacco dono consistente in vari prodotti della Systems Editoriale (riviste e dischetti).

```
Rem   Applicazione del Teorema di Pitagora
Rem  Programma scritto da Enrico Marchesin
Rem                  con Amiga AMOS
Rem ****************************************
SCRW=320 : SCRH=256 : CO=16 : SCRMODE=Lowres
Screen Open 0,SCRW,SCRH,CO,SCRMODE
Ink 2 : Paper 0 : Change Mouse 2 : Flash Off
Cls : AC=40 : AB=30
TDRAW[AC,AB,L]
While Inkey$<>""
   XX=X Screen(X Mouse) : YY=Y Screen(Y Mouse)
If Mouse Key=1 Then SETL[XX,YY]
Wend
End
Procedure TDRAW[C1,C2,L]
   Global SCRW,SCRH,X,Y
   Cls : DSCRW=SCRW : DSCRH=SCRH-32
   TW=2*C1+C2
   TH=C1+2*C2
   If DSCRW<TW Then PAR=1 : Goto FINP
   If DSCRH<TH Then PAR=2 : Goto FINP
   X=Int(C1+(DSCRW-TW)/2)
   Y=Int(C1+C2+(DSCRH-TH)/2)
   Box X-C1,Y-C1 To X,Y
   Box X,Y To X+C2,Y+C2
   Polyline X,Y-C1 To X+C2,Y To X+C1+C2,Y-C2 To
X+C1,Y-C1-C2 To X,Y-C1
   Ink 4 : On L+1 Gosub FINP,CAT1,CAT2
   Ink 2 : Goto FINP
   CAT1:
   Draw X,Y To X,Y-C1
   Return
   CAT2:
   Draw X,Y To X+C2,Y
   Return
   FINP:
   Locate 0,30
   Centre "     AC="+Str$(C1)+"  AB="+Str$(C2)+"
BC="+Str$(Sqr(C1^2+C2^2))+"      "
   If L=0
      Locate 0,31
      Centre "ACGF="+Str$(C1^2)+"
ADEB="+Str$(C2^2)+" CBIH="+Str$(C1^2+C2^2)
   End If
End Proc[PAR]
Procedure SETL[MX,MY]
   Global AC,AB,X,Y,SCRW,SCRH
   If MX>X-5 and MX<X+5 and MY>Y-AC-5 and MY<Y+5
Then L=1
   If MX>X-5 and MX<X+AB+5 and MY>Y-5 and MY<Y+5
Then L=2
   If L=0 Then Goto FINE
   TDRAW[AC,AB,L]
   While Mouse Key=1
      MX=X Screen(X Mouse) : MY=Y Screen(Y Mouse)
      If L=1 and MY<Y
         AC=Y-MY
         TDRAW[AC,AB,L]
      Else
         If MX>X
            AB=MX-X
            TDRAW[AC,AB,L]
         End If
      End If
   Wend
   While Param<>0
      AC=AC-1 : AB=AB-1
      TDRAW[AC,AB,L]
   Wend
   FINE:
   L=0
   TDRAW[AC,AB,0]
End Proc
```

Il velocissimo programma in Amos utilizza le procedure; le righe sottolineate vanno digitate su un unico rigo.

```
' Teorema di Pitagora
' Versione per AMIGA in AMOS
' By  Simone Riccetti
INIZIALIZZAZIONE:
 Screen Open 1,640,256,4,Hires
 Paper 0 : Cls : Curs Off : Change Mouse 2
 For K=1 To 5
  Reserve Zone K
 Next K
 Set Zone 1,560,150 To 582,166
 Set Zone 2,535,166 To 560,176
 Set Zone 3,560,166 To 581,176
 Set Zone 4,581,166 To 608,176
 Set Zone 5,560,176 To 581,192
 BX=230 : BY=120 : AB=15 : BC=15
CALCOLA:
 AX=BX : AY=BY-AB : CX=BX+(BC*2) : CY=BY
 X1=AX+(AB*2) : Y1=AY-BC : X2=CX+(AB*2) :
Y2=CY-BC
 X3=CX : Y3=CY+BC : X4=BX : Y4=BY+BC
 X5=BX-(AB*2) : Y5=BY : X6=AX-(AB*2) : Y6=AY
 AC#=Sqr((AB*AB)+(BC*BC))
CONTROLLA:
 If X2>639 and X5>1 Then BX=BX-1 : FLAGX=1 :
Goto CALCOLA
 If Y1<1 and Y4<256 Then BY=BY+1 : FLAGY=1 :
Goto CALCOLA
 If FLAGX=0 and BX<230 and X2<639 Then BX=BX+1 :
Goto CALCOLA
 If BX>320 Then BX=320 : Goto CALCOLA
 If FLAGY=0 and Y1>1 and BY>120 Then BY=BY-1 :
Goto CALCOLA
 If BY<120 Then BY=120 : Goto CALCOLA
 FLAGX=0 : FLAGY=0
DISEGNA:
 Ink 2,0,0 : Gosub TRACCIA
 Locate 2,1 : Pen 1 : Print "AREE DEI QUADRATI"
 Pen 2 : Print Using "AB = ######";AB^2
 Print Using "BC = ######";BC^2
 Print Using "AC = ######";AC#^2
 Locate 2,20 : Pen 1 : Print "LUNGHEZZA LATI"
 Pen 2 : Print Using "AB = ####.00";AB
 Print Using "BC = ####.00";BC
 Print Using "AC = ####.##";AC#
 Pen 2 : Locate 71,21 : Print "E"
 Pen 1 : Locate 71,19 : Print "A"
 Locate 71,23 : Print "V"
 Locate 68,21 : Print "<"
 Locate 74,21 : Print ">"
 Ink 1,0,0
 Box 560,150 To 582,192
 Box 535,166 To 608,176
 Ink 0,0,0 : Draw 571,156 To 572,156
RICEVE_COMANDO:
 While Mouse Key<>1
 Wend
 X=Mouse Zone
 If X=0 Then Goto RICEVE_COMANDO
 Ink 0,0,0 : Gosub TRACCIA
 If X=2 Then BC=BC-1
 If BC<1 Then BC=1
 If X=4 Then BC=BC+1
 If X=1 Then AB=AB+1
 If X=5 Then AB=AB-1
 If AB<1 Then AB=1
 If X=3 Then Cls : End
 Ink 2,0,0 : Goto CALCOLA
TRACCIA:
 Draw AX,AY To BX,BY : Draw BX,BY To CX,CY
 Draw CX,CY To AX,AY : Draw AX,AY To X1,Y1
 Draw X1,Y1 To X2,Y2 : Draw X2,Y2 To CX,CY
 Draw BX,BY To X4,Y4 : Draw X4,Y4 To X3,Y3
 Draw X3,Y3 To CX,CY : Draw BX,BY To X5,Y5
 Draw X5,Y5 To X6,Y6 : Draw X6,Y6 To AX,AY
 Text AX-9,AY-2,"A"
 Text BX-9,BY+7,"B"
 Text CX+4,CY+7,"C"
 Return
```

Il programma di Simone Riccetti; le righe sottolineate devono esser digitate su un unico rigo.

Un problema, che capita più spesso di quanto non si creda, riguarda al corretta suddivisione di un rettangolo in più parti

Diamoci un taglio!

⇨ < Una sfida al mese > — < E' la tradizionale sfida mensile lanciata a tutti i lettori! >

Anche questo mese la tradizionale sfida riguarda un problema grafico, che riassumiamo brevemente:

Supponiamo di avere a disposizione un rettangolo (ad esempio un foglio di carta) di dimensioni **X** (orizzontale) ed **Y** (verticale). **Vogliamo ottenere il maggior numero possibile di due gruppi di rettangoli di dimensioni minori X_1, Y_1 (il primo) ed X_2, Y_2 (il secondo).**

Il problema, posto in altro modo, è quello di ottenere il minor scarto possibile.

Vediamo di chiarire: nel caso in cui, ad esempio, si volessero ottenere, da un foglio di grandi dimensioni, rettangoli più piccoli di un'unica grandezza (vedi figura **1** e **2**) la disposizione **orizzontale** oppure **verticale** della dimensione maggiore consente di realizzare un diverso numero di rettangoli minori.

Nel caso in cui, poi, si volessero ottenere due tipi di rettangoli di dimensioni diverse tra loro (vedi figura 3) le cose si possono complicare notevolmente.

Anche nel caso in cui si volessero ottenere rettangoli minori di dimensione unica una particolare disposizione, apparentemente disordinata, potrebbe invece portare all'acquisizione di un numero di figure maggiore che nel caso di una suddivisione "ordinata".

La sfida

Non pensate che la soluzione sia semplice. Si tratta, infatti, di realizzare una routine ripetitiva che tenga conto di una miriade di casi possibili (ma tutti, come intuitivo, parametrizzabili tenendo conto di alcune "condizioni al contorno").

In parole povere, il programma, appena lanciato, deve più o meno funzionare così (in *corsivo* le domande poste dal programma, in **neretto** le risposte dell'utente):

Dimensione X foglio? **190**
Dimensione Y foglio? **100**
Dimensione lato X rett.? **22**
Dimensione lato Y rett.? **37**

In seguito dovrebbe apparire sul video, in alta risoluzione, una figura simile a quelle pubblicate in queste pagine.

Ci accontentiamo, quindi, di un programma che sia in grado di elaborare la suddivisione ottimale di una sola dimensione (fig. 1 e 2); i **più bravi** potranno tentare la suddivisione di due o più rettangoli di dimensione diversa (fig. 3); in tal caso dovrebbero esser poste altre domande:

Dimensione lato X rett. 2? **12**
Dimensione lato Y rett. 2? **28**

...e fornire, in risposta, la figura 3 (o similari).

Pensate che sia semplice? Beh, almeno provateci!

All'autore del programma prescelto verrà inviato un **completo set di efficienti accendisigari tascabili** (specificare se si desidera una scatola di *Cerini* oppure di *Minerva*).

Salvo, ovviamente, esaurimento scorte...

Come partecipare

Nelle sfide mensili viene sempre data precedenza ai programmi, scritti in qualsiasi linguaggio, in grado di girare su **Amiga** oppure computer **Ms - Dos** compatibili. I listati devono essere strutturati in modo chiaro, allo scopo di favorire eventuali modifiche da parte di lettori. Listati ed articolo esplicativo devono *tassativamente* essere inviati su disco (qualsiasi formato) oppure, via modem, alla nostra BBS (tel. 02/57.60.52.11). Onde evitare invio di materiale inadeguato alla pubblicazione, si consiglia di telefonare in redazione (Tel. 02/57.60.63.10) per concordare l'inoltro del lavoro svolto.

1 6

12

Suddivisione di un rettangolo di cm. 16.8 x 5.4 in due file di rettangoli di cm. 2.7 x 2.1 per un totale di 12 mini rettangoli.

1 8

16

Le dimesioni del rettangolo grande e di quelli piccoli sono le stesse della figura precedente. In questo caso, però, si ottengono 16 mini rettangoli.

1 a 8 a

16 a

1 b 8 b

Qui il problema è di ottenere due dimensioni diverse. Questo schema di taglio potrebbe non essere il migliore:
è quindi necessario scrivere un programma in grado di ottimizzare i tagli da effettuare.

Un contatempo in Turbo Pascal

< di Andrea Eugenio Nalmoli >

< Esaminare C. C. n. 79 >

Ancora con quel programma? Ma da quanto tempo ci sei sopra?

Chissà quante volte ci si è sentiti rivolgere una domanda del genere e non si è saputo rispondere! Ci vorrebbe proprio un bel programmino in grado di gestire i tempi di utilizzo di altri programmi...

Il programma

Il listato di queste pagine risponde alla sfida lanciata nel numero **79** e, da un punto di vista didattico, svolge una funzione interessante dal momento che illustra un modo poco di programmare in Pascal.

La ***sfida*** in questione chiedeva di **realizzare un programma in grado di archiviare i tempi di utilizzo di un qualunque file eseguibile** e proponeva un metodo basato sulla memorizzazione sequenziale dei tempi e sulla successiva interpretazione da parte di un programma apposito.

Il programma svolge tutte le operazioni necessarie: *memorizzazione* dei tempi, *determinazione* del tempo trascorso, *gestione* del file-dati associato. Una caratteristica importantissima è che il file-dati contiene sempre e solo il tempo totale di utilizzo (nel formato giorni - ore: minuti: secondi) e perciò non si espande continuamente rubando bytes preziosi al disco rigido. Tale particolarità permette, inoltre, di leggerne il contenuto impartendo un banale `TYPE filename.ext` direttamente a livello di comandi DOS.

Commenti

Il listato contiene alcune rem nei punti più significativi. Sono da notare le direttive al compilatore **{$M 8192, 0, 0}**, **{$I-}** e **{$I+}** la cui funzione è relativamente complessa: basterà dire che risultano necessarie per la buona funzione di alcuni comandi presenti nel listato (come quelli di accesso al disco).

Si possono naturalmente migliorare le prestazioni offerte, ma si può anche alleggerire ulteriormente il listato a seconda delle esigenze (potrebbe non servire la visualizzazione finale dei dati, per esempio); la versione viene presentata in una forma idonea per una presentazione di tipo prevalentemente didattico. Il formato usato per il tempo può essere ridotto alle sole ore e minuti, ma questa libertà viene lasciata al lettore.

Come operare

Dopo aver caricato il main-file del Turbo-Pascal, si deve digitare il listato e compilarlo su disco con il nome ART (un qualsiasi altro nome, comunque, andrà bene egualmente).

Per quanto riguarda i comandi dell'interprete-compilatore Turbo-Pascal si rimanda al manuale o al comodissimo Help interno.

Digitando, dopo la compilazione (e, ovviamente, dopo essere "usciti" da T. Pascal) ART, privo di parametri, verrà fornita la sintassi del comando:

ART filename.ext [args]

...in cui **filename.ext** è il file-programma da eseguire e **args** sono gli eventuali parametri di cui lo stesso programma da lanciare abbisogna.

Naturalmente l'utente dovrà essere a conoscenza di tali parametri dal momento che il programma non fa altro che lanciarlo "arricchendolo" dei parametri che verranno digitati in coda al comando stesso.

Quando "uscirete" dal programma, verrà visualizzato il tempo trascorso ed il tempo globale di utilizzo, mentre il file-dati avrà nome **filename.ART**.

Il suffisso è stabilito a priori (vedi listato) ma sarà sufficiente modificarlo assegnadogli una qualsiasi terna di caratteri leciti.

Peculiarità

Tutte le operazioni sono svolte dall'unico file programma generato in fase di compilazione.

Il file da eseguire può essere il nome di un programma commercializzato, come ad esempio **Word/Z** (Word comprensivo di cursore lampeggiante), **WS5** (Word Star 5), **QB** (Quick Basic MicroSoft), un file batch (**VPprof**: Ventura) o addirittura un comando interno del DOS!

Il file-dati è analizzabile con il semplice comando Type dal momento che è in formato ASCII e, come tale, modificabile a piacimento, anche manualmente.

Da notare che il file con suffisso .ART verrà "depositato" nel drive e nella directory in cui è presente il programma lanciato. Se, pertanto, tale programma è posizionato in una sua directory, all'interno di questa si troverà il file Nome.ART.

Se, però, il programma in oggetto consente, durante il suo funzionamento, di uscire dalla directory di lavoro, questa potrebbe essere sempre diversa a seconda del momento in cui si decide di uscire dal programma. Ne consegue che potrebbe esserci più di un file Nome.ART disseminati in altrettanti dischi e/o direc-

Il listato sorgente in Turbo Pascal di A. E. Naimoli. Altri lettori, in precedenza, avevano inviato una soluzione al problema proposto sul n. 79, ma questa è sicuramente la più semplice e valida di tutte grazie alla sua brevità ed al fatto che l'intera procedura è contenuta in un unico file.

```
program ART;
{by Andrea Eugenio Naimoli - Activate Recording Time [RUN 3.0]}

{ INIZIALIZZAZIONE VARIABILI }

{$M 8192,0,0}
uses crt,Dos;
var args,c,cd,ch,cm,cs,name:string;
    a:integer;
    h,m,s,d,h1,m1,s1:word;
    ss,se,t,h2,m2,s2,day:longint;
    f:text;

{ LA PROCEDURA Esci(code) RESTITUISCE UN CODICE DI ERRORE }

procedure Esci(e:byte);
begin
 case e of
 1:writeln('Usage: ART filename.ext [args,...]');
 2:writeln('Error #2 - COMMAND.COM not executable')
 end;
 halt(e)
end;

BEGIN
 writeln;writeln('Andrea Eugenio Naimoli presents ART');writeln;

{ SE CI SONO DEI PARAMETRI LI ACCODA AL NOME DEL FILE }

 if paramcount=0 then Esci(1);
{ La prossima riga, sottolineata, va digitata su un unico rigo!}
 if paramcount>1 then for a:=2 to paramcount do args:=args+'
'+paramstr(a);
 c:='/c '+paramstr(1)+args;

{ LEGGE IL TEMPO INIZIALE E LO MEMORIZZA TUTTO IN SECONDI }
```

```
 gettime(h,m,s,d);h2:=h;m2:=m;s2:=s;ss:=s2+m2*60+h2*3600;

{ ESEGUE IL PROGRAMMA RICHIAMATO }

 SwapVectors;
 Exec(GetEnv('COMSPEC'), c);
 SwapVectors;

{ LEGGE IL TEMPO FINALE E CALCOLA IL TEMPO TRASCORSO }

 gettime(h,m,s,d);h2:=h;m2:=m;s2:=s;se:=s2+m2*60+h2*3600;
 se:=se-ss;
 h2:=se div 3600;h1:=h2;
 m2:=(se-h1*3600) div 60;m1:=m2;
 s2:=se-h1*3600-m1*60;s1:=s2;
 if DosError <> 0 then Esci(2);

{ IL FILE-DATI E' IL NOME DEL FILE ESEGUITO + .ART}

 a:=pos('.',paramstr(1));
 if a=0 then name:=paramstr(1)+'.art'
 else name:=copy(paramstr(1),1,a-1)+'.art';

{ VERIFICA LA PRESENZA DEL FILE-DATI }
 {$I-}
 assign(f,name);
 reset(f);
 close(f);
 {$I+}

{ SE CI SONO DATI MEMORIZZATI LI LEGGE }

 if ioresult=0 then begin
  reset(f);
  read(f,c);
  close(f);
  cd:=copy(c,1,pos('-',c)-1);
  ch:=copy(c,pos('-',c)+1,pos(':',c)-pos('-',c)-1);
  cm:=copy(c,pos(':',c)+1,length(c));
  cs:=copy(cm,pos(':',cm)+1,length(cm));
  cm:=copy(cm,1,pos(':',cm)-1);
  val(cd,day,a);
  val(ch,h2,a);
  val(cm,m2,a);
  val(cs,s2,a);
  ss:=s2+m2*60+h2*3600+day*86400
 end
 else ss:=0;

{ CALCOLA IL TEMPO TOTALE (ATTUALE+MEMORIZZ.) E MOSTRA I DATI }

 t:=ss+se;day:=t div 86400;h2:=(t-day*86400) div 3600;
 m2:=(t-day*86400-h2*3600) div 60;s2:=t-day*86400-h2*3600-m2*60;
 writeln('Actual time is - ',h1:2,':',m1:2,':',s1:2);
 writeln('Total time is - ',day:2,'-',h2:2,':',m2:2,':',s2:2);

{ MEMORIZZA I NUOVI DATI SUL FILE-DATI }

 rewrite(f);
 writeln(f,day,'-',h2,':',m2,':',s2);
 close(f);
 writeln
END.
```

tory. Per evitare questo, vi consigliamo di attivare sempre un file batch in modo da esser sicuri di ritrovarsi in una ben precisa directory.

Ad esempio, per attivare il word processor Word MicroSoft, posizionato per ipotesi nella directory **Word** del disco **d**, usate il seguente file batch di nome `Word_tem.bat` e posizionatelo nella root di **c**:

```
d:
cd\
cd word
word/z
c:\
```

Ora digitate il seguente file batch...

```
c:tempi\tempi_pr.exe
word_tem.bat
```

...di nome `Word.bat` e posizionatelo nella root di c. Ipotizziamo, ovviamente, che il programma compilato di queste pagine sia stato memorizzato in **c:** directory **Tempi** con il nome **Tempi_pr.exe** (e non **Art** come indicato all'inizio dell'articolo).

Digitando semplicemente **word** (supponendo di essere nella root di c:), e premendo il tasto Return, alla fine di ciascuna sessione di lavoro con Word avrete sempre aggiornato, nel file **c: word_tem.art**, il tempo totale di utilizzo del word processor.

Concludendo

Se occorre, potete modificare il codice presentato, lasciando possibilmente inalterate le note di "copyright" (un po' di orgoglio non guasta...).

Ricordate che, se avete un **Amiga** con scheda emulatrice IBM priva di hard disk, potete ugualmente far funzionare il Turbo-Pascal adoperando un drive virtuale (comando: **Jlink**) allocato nella Ramdisk dell'Amiga che contenga i file **Turbo.Exe**, **Turbo.Tpl**, **Thelp.Com** e lasciando il drive **A:** disponibile.

Non dovrebbero esserci particolari problemi scrivendo il programma in **Kick Pascal**, apportando, magari, alcune modifiche che l'utente esperto sarà sicuramente in grado di effettuare da solo.

Due giochi molto popolari sono un'ottima scusa per realizzare un'implementazione sui computer più diffusi tra gli appassionati.

Due giochi per C/64, Amiga, Ms-Dos

< di Dario Pistella>

< Esaminare C. C. n. 85: "Un gioco truccato..." e "Giochiamo a Briscola" >

Il programma più breve presentato in queste pagine è l'implementazione di un gioco molto semplice, che si gioca di solito su un foglio di carta, *simile* al più diffuso gioco del **Tris**.

Delle due versioni, la prima è valida per **AmigaBasic**; la seconda, invece, gira allo stesso modo sia con il **Gw-Basic** oppure **QuickBasic** (sistemi Ms-Dos), sia con il **C/64**, purché dotato, ovviamente, di *emulatore* Gw-Basic.

Può esser considerato del tutto simile al "*Binary Game*", già pubblicato sul N. **85**; offre, però, il vantaggio di essere molto più breve da digitare e rappresenta un approccio diverso alla soluzione del problema.

Come si gioca

Le regole sono molto semplici: su un numero variabile di linee (da una a nove), vengono disposte delle palline, anche queste in numero variabile (da una a dieci). Due giocatori, a turno, possono cancellare un numero a piacere di palline, ma su una sola linea alla volta. Il giocatore che sarà costretto a cancellare l'ultima pallina, avrà perso.

Il primo programma - gioco

Il funzionamento del programma è identico sia nella versione AmigaBasic che per Gw-Basic.

Per prima cosa verrà richiesto il numero di linee sulle quali si intende giocare, che saranno disegnate (in bassa risoluzione) subito dopo.

Accanto ad ognuna di esse, a sinistra, sarà visualizzato il numero d'ordine con cui appaiono, mentre a destra appariranno i numeri di palline presenti su ciascuna linea (vedi figura), generati in modo casuale.

Per indicare all'elaboratore la mossa che si intende fare, sarà sufficiente digitare il numero della linea su cui si intende operare, seguito da una virgola e dal numero di palline che si desidera cancellare.

Dopo aver eseguito l'operazione richiesta il computer risponderà, a sua volta, segnalando anche graficamente la mossa effettuata.

Alla fine del gioco, apparirà un messaggio che indicherà il vincitore.

Il programma è molto semplice ma, come sempre accade per i programmi che implicano una risposta da parte dell'elaboratore, si può migliorare la routine relativa al ragionamento del computer.

Nella versione pubblicata, la routine è basata su un algoritmo che si occupa di lasciare il giocatore avversario in condizione di avere a disposizione un numero totale di palline (variabile TOT*) che sia pari se il numero di linee rimaste (variabile* TOL*) è pari, dispari se il numero di linee è dispari.*

Sono anche analizzati i casi in cui le colonne rimaste siano due (TOL = 2) oppure una sola.

```
1 - ********** 10
2 - *******     7
3 - ****        4
4 - *****       5
5 - ******      6
6 - ******      6
7 - *********   9
8 - **          2
9 - *           1
```

Esempio di schema del gioco

Anche se, tramite questo algoritmo, tutti i casi possibili sono contemplati e quindi niente (o quasi) viene lasciato al caso, vi sono comunque casi che dovrebbero essere trattati singolarmente e che necessitano di particolari risposte.

Migliorie suggerite

I programmi, come ogni listato che si rispetti, sono suscettibili di variazioni ed ampliamenti di vario genere, tra cui:

➪ una miglior resa grafica (magari con sprites o altre piacevolezze);

➪ miglioramento della routine che gestisce la risposta dell'elaboratore;

➪ annullamento dell'ultima mossa;

➪ richiesta di suggerimento;

➪ e così via, all'infinito.

A voi, quindi, il compito di migliorare il programma usando tutta la vostra bravura di programmatori.

```
REM Righe! (Versione AmigaBasic)
REM by Dario Pistella
INPUT "Linee ";a$:a = VAL(a$)
IF a < 1 OR a > 9 THEN RUN
CLS:RANDOMIZE TIMER
FOR cic=1 TO a:pal(cic) = INT(10*RND)+1:tot = tot + pal(cic)
  PRINT cic"- ";STRING$(pal(cic),42);TAB(18)pal(cic)
NEXT:tol = a
Giocatore:
gi = 1:LOCATE 15,1:PRINT SPACE$(30)
LOCATE 15,1:INPUT "Linea, palline";b,c
  IF b < 1 OR b > a THEN Giocatore
  IF c < 1 OR c > pal(b) THEN Giocatore
pal(b) = pal(b) - c:GOSUB cancella
Computer:
gi = 2:b = 0:d = 0:x = 0
FOR cic = 1 TO a
  IF pal(cic) > 2 THEN b = cic
  IF pal(cic) = 2 THEN d = cic
  IF pal(cic) = 1 THEN u = cic
NEXT:IF b AND tol > 2 THEN sort
IF tol = 1 THEN b = b + u + d:c = pal(b) - 1:GOTO out
GOSUB stat:c = 1:IF b = 0 THEN b = d
  IF tol = 2 AND u > 0 AND b > 0 THEN c = pal(b):GOTO out
  IF tol = 2 AND b > 0 AND b <> d THEN c = pal(b) - 2
  IF b = 0 OR (b > 0 AND tok = toi AND tol <> 2) THEN b = u
out:
GOTO stampa
sort:
c=INT( (pal(b)-1) *RND)+1:GOSUB stat
  IF toi = tok AND ci = 0 THEN sort
  IF toi <> tok AND ci = 1 THEN sort
stampa:
pal(b) = pal(b) - c
LOCATE 14,1:PRINT "Computer:":PRINT TAB(15)"  "b","c
FOR k=1 TO 6000:NEXT:GOSUB cancella:LOCATE 14,1:PRINT SPACE$(10)
GOTO Giocatore
cancella:
LOCATE b,6:PRINT STRING$(pal(b),42);SPACE$(10-pal(b));TAB(18)pal(b)
  IF pal(b) = 0 THEN tol = tol - 1
tot = tot - c:IF tot = 1 THEN Fine
RETURN
stat:
ci = 0:toi = 0:tok = 0
  IF tol/2 = INT(tol/2) THEN toi = 1
  IF tot/2 = INT(tot/2) THEN tok = 1
  IF c/2 = INT(c/2) THEN ci = 1
RETURN
Fine:
CLS:IF gi = 1 THEN PRINT "Hai vinto!" ELSE PRINT "Ho vinto!"
```

Il brevissimo programma che implementa il gioco.

Giochiamo a scopa

Il secondo gruppo di programmi che presentiamo in queste pagine, come sempre nella duplice versione **Amiga / Gw-Basic**, non è altro che l'implementazione di uno dei più famosi e diffusi giochi di carte: la scopa.

Per quei pochi che non dovessero conoscere **le regole**, ricordiamo che:

➪ si utilizza un mazzo di carte italiane (**D**anari, **S**pade, **B**astoni, **C**oppe);

➪ si distribuiscono **tre** carte a ciascun giocatore, e nella prima mano, **quattro** carte devono essere scoperte e messe in tavola;

➪ con una delle proprie carte si possono prendere carte presenti sul tavolo in modo tale che la loro somma sia uguale al valore della carta utilizzata;

➪ se, e solo se, non c'è alcun modo di prendere carte, una carta può essere lasciata sul tavolo.

Per quanto riguarda i **punteggi**, ricordiamo che se si esauriscono le carte presenti sul tavolo, cioè si è realizzata una scopa, questa vale un punto; inoltre il Settebello (sette di danari) vale un punto come pure il maggior numero di danari ed il maggior numero di carte prese.

La primiera si calcola, per ogni giocatore, in base alla somma dei valori delle carte migliori di ogni seme.

Questi sono i valori delle carte utilizzati nel calcolo della primiera (rispettivamente: carta, valore): **7**, 21 - **6**, 18 - **5**, 15 - **4**, 14 - **3**, 13 - **2**, 12 - **1**, 16 - **8**, 10 - **9**, 10 - **10**, 10.

Il programma

Dopo aver lanciato il programma, sullo schermo appariranno, in alto a sinistra, le vostre tre carte; in centro le quat-

```
a = st:GOSUB cala:IF x <> 2 THEN caso
x = 0:IF sbb AND (v(2,sco) = 7 OR sco = 0) THEN a = sbb
prencom:
FOR se=1 TO t:a$=a$+RIGHT$(STR$(t(se)),1):NEXT:GOTO giocprende
caso:
FOR se=1 TO 3:IF v(2,se) <> 0 THEN a = se
NEXT:GOSUB cala:IF x = 2 THEN prencom
GOSUB pr:GOTO ancora
cala:
x=0:FOR cic=1 TO vt:vv(cic) = vt(cic):p(cic) = cic:NEXT
ord:
FOR cic=1 TO vt-1:IF vv(cic) >= vv(cic+1) THEN ok
h = vv(cic+1):vv(cic+1) = vv(cic):vv(cic) = h
h = p(cic+1):p(cic+1) = p(cic):p(cic) = h:GOTO ord
ok:
NEXT
FOR cic=1 TO vt:v = v(gioc,a):IF x <> 2 THEN t = 0
 FOR u=cic TO vt
  IF x <> 2 AND (v - vv(u)) > -1 THEN v=v-vv(u):t=t+1:t(t)=p(u)
  IF v = 0 THEN x = 2
 NEXT u,cic
IF gioc = 2 THEN RETURN
IF x = 2 THEN LOCATE 6,1:nv=1:PRINT "NON VALIDO":RETURN
pr:
vt = vt + 1:vt(vt)=v(gioc,a):st(vt)=s(gioc,a):LOCATE 15,(6*vt)
PRINT vt:PRINT TAB(6*vt);v(gioc,a);a$(st(vt))
canc:
IF gioc = 1 THEN LOCATE a,5:PRINT SPACE$(6)
RETURN
vase:
valore=INT(10*RND)+1:segno=INT(4*RND)+1
IF check(valore,segno)=1 THEN vase
check(valore,segno) = 1:RETURN
fine:
gioc = up
FOR cic=1 TO vt:vct=vt(cic):sct=st(cic):GOSUB valutcart:NEXT
CLS:PRINT TAB(14)"Giocatore:";TAB(34)"Computer:"
PRINT "Scope";TAB(18);sc(1);TAB(38);sc(2)
PRINT "Settebello";TAB(20*(sbp-1)+19)"*"
PRINT "Carte";TAB(18);ct(1);TAB(38);ct(2)
IF ct(1)>20 THEN pu(1)=pu(1)+1 ELSE IF ct(1)<20 THEN pu(2)=pu(2)+1
FOR gio=1 TO 2
FOR cic=1 TO 4:vx=0:mx(gio,cic)=0:FOR cic2=1 TO ct(gio)
IF srt(gio,cic2)<>cic THEN nosegn
vx = crt(gio,cic2) + 10:IF vx > 17 THEN vx=10
IF crt(gio,cic2) = 7 OR crt(gio,cic2)=6 THEN vx=crt(gio,cic2)*3
IF crt(gio,cic2) = 1 THEN vx = 16
IF vx > mx(gio,cic) THEN mx(gio,cic) = vx
IF cic = 1 AND srt(1,cic2)=1 THEN dan(1)=dan(1)+1
nosegn:
NEXT cic2,cic,gio
FOR cic=1 TO 2:FOR cic2=1 TO 4
p(cic) = p(cic) + mx(cic,cic2):NEXT cic2,cic
tb=29:IF p(1) > p(2) THEN pu(1) = pu(1) + 1:tb=19
IF p(2) > p(1) THEN pu(2) = pu(2) + 1:tb=39
PRINT "Danari:";TAB(18);dan(1);TAB(38);10-dan(1)
PRINT "Primiera:";TAB(tb)"*":PRINT
IF dan(1) > 5 THEN pu(1) = pu(1) + 1
IF dan(1) < 5 THEN pu(2) = pu(2) + 1
PRINT "Punti:";TAB(18);pu(1);TAB(38);pu(2)
```

N.B.: Alcune righe, per esigenze di impaginazione, sono state stampate in caratteri più piccoli allo scopo di farle rientrare in una sola riga.

tro carte lasciate in tavola ed infine il numero di carte rimaste nel mazzo.

Per comunicare la carta che si intende utilizzare, sarà sufficiente digitare il **numero** relativo alla carta in questione.

Quindi, alla richiesta delle carte che si intende prendere, basterà inserire, in successione e *senza* spazi, i numeri relativi alle carte presenti sul tavolo.

Se, invece, si vuole lasciare in tavola la carta, bisognerà digitare la lettera **L** (lasciata).

In base alle routines presenti nel programma, l'elaboratore scriverà il messaggio non valido nei casi in cui:

⇨ la somma delle carte richieste non sia uguale al valore della carta che si intende utilizzare;

⇨ non si possa lasciare la carta perchè in tavola quella stessa carta potrebbe prenderne altre (ricordiamo infatti che se c'è la possibilità, è obbligatorio prendere);

⇨ si vogliano prendere più carte il cui valore totale sia uguale a quello di una singola carta presente in tavola (es: se si vuole usare un **8** ed in tavola vi sono le carte **7, 1, 8**, non si potranno prendere le prime due carte, ma si dovrà prendere la terza).

Dopo essersi accertato che l'operazione richiesta sia valida, il programma si occuperà della gestione grafica dello schermo e valuterà le carte prese e quella usata, aggiornando, se necessario, il punteggio.

Come gioca il computer

La routine di risposta da parte del computer si occupa soltanto di controllare se, tra le sue carte, vi sia la possibilità di realizzare una scopa, di prendere il settebello o un qualsiasi altro sette.

Se nessuna di queste possibilità si verifica, la carta da utilizzare viene scelta casualmente.

Le strategie di gioco possono essere le più svariate e sta quindi a voi inserire una routine che lasci il minor spazio possibile alla casualità.

Per quanto riguarda le **variabili utilizzate** nel corso del programma, si ricordi che la **V** (giocatore, numero carta) e la **S** (giocatore, numero carta) contengono, rispettivamente, il valore ed il segno delle

```
REM Scopa; Versione AmigaBasic
REM by Dario Pistella
RANDOMIZE TIMER
DIM crt(2,40),srt(2,40)
a$(1) = "d":a$(2) = "b":a$(3) = "s":a$(4) = "c":vt = 4:CLS
docarte:
IF ca=40 THEN fine
FOR dare=1 TO 3:FOR g=1 TO 2:GOSUB vase
  IF g = 1 THEN LOCATE dare,1:PRINT dare"- "valore;a$(segno)
  v(g,dare) = valore:s(g,dare) = segno
NEXT g,dare:ca = ca + 6:IF ca = 6 THEN ca = 10
LOCATE 19,1:PRINT"Rimaste:";40 - ca
IF ca > 10 THEN RETURN
FOR tav=1 TO 4:GOSUB vase
  LOCATE 15,(6*tav):PRINT tav:PRINT TAB(6*tav);valore;a$(segno)
  vt(tav) = valore: st(tav) = segno:NEXT
attesa:
gioc = 1:nv = 0:x = 0:va = 0:a$ = INKEY$
a = VAL(a$):IF a < 1 OR a > 3 OR v(1,a) = 0 THEN attesa
LOCATE 5,1:PRINT v(1,a);a$(s(1,a))
PRINT SPACE$(20):LOCATE 6,1:INPUT "carte ";a$
IF a$ = "1" THEN GOSUB cala:GOTO ancora
FOR con=1 TO vt:vh = VAL(MID$(a$,con,1))
  IF vh <= vt THEN va=va + vt(vh)
  IF vt(con) = v(1,a) AND LEN(a$) > 1 THEN x = 1
NEXT
IF (va <> v(1,a)) OR x THEN
LOCATE 6,1:PRINT "NON VALIDO":nv = 1:GOTO ancora
END IF
giocprende:
cox=0:x=0:LOCATE 5,1:PRINT v(gioc,a);a$(s(gioc,a))
PRINT "carte:"a$"    "
FOR cic=1 TO 2000:NEXT
LOCATE 15,1:PRINT SPACE$(80):PRINT SPACE$(80)
FOR con=1 TO vt:y = 0
 FOR con2=1 TO LEN(a$):co = VAL(MID$(a$,con2,1))
  IF co = con THEN y=1:vct=vt(con):sct=st(con):GOSUB valutcart
 NEXT:IF y THEN noprint
 cox=cox+1:LOCATE 15,(6*cox):PRINT cox
PRINT TAB(6*cox);vt(con);a$(st(con))
 vt(cox) = vt(con):st(cox) = st(con)
noprint:
NEXT:vt=vt-LEN(a$):vct=v(gioc,a):sct=s(gioc,a):GOSUB valutcart
IF vt = 0 THEN pu(gioc)=pu(gioc)+1:sc(gioc)=sc(gioc) + 1
IF gioc = 1 THEN GOSUB canc
up = gioc:GOTO ancora
valutcart:
IF sct = 1 AND vct = 7 THEN pu(gioc) = pu(gioc) + 1:sbp = gioc
ct(gioc)=ct(gioc)+1:crt(gioc,ct(gioc))=vct:srt(gioc,ct(gioc))=sct
RETURN
ancora:
IF nv=0 THEN v(gioc,a)=0:s(gioc,a)=0
FOR u=1 TO 5000:NEXT:LOCATE 5,1:PRINT SPACE$(20):PRINT SPACE$(20)
IF v(1,1)+v(2,1)+v(1,2)+v(2,2)+v(1,3)+v(2,3)=0 THEN GOSUB docarte
IF nv OR gioc=2 THEN attesa
Computer:
summ=0:FOR sum=1 TO vt:summ=summ+vt(sum):NEXT
a$ = "":gioc = 2:st = 0:sbb = 0:sco = 0:REM contr 7,7d,scopa
FOR se=1 TO 3
IF v(2,se) = 7 THEN st=se:IF s(2,se) = 1 THEN sbb = se
IF v(2,se) = summ AND summ <> 0 THEN sco = se
NEXT:IF sco THEN st = sco
IF st = 0 THEN caso
```

carte dei due giocatori, mentre in **VT** (numero carta) e **ST** (numero carta) vi sono, rispettivamente, il valore ed il segno delle carte in tavola; infine la variabile **PU** (numero giocatore), contiene il totale dei punti totalizzati dai giocatori.

Come potrete notare da un'attenta analisi del listato per Amiga, vi sono alcune subroutines che vengono chiamate sia dalle routines che gestiscono le mosse del giocatore, sia da quelle che si occupano, invece, delle decisioni prese dall'elaboratore. Un meccanismo interessante è quello della routine **Cala**: in questa fase la carta prescelta da uno dei giocatori viene analizzata per controllare se con essa sia possibile, o meno, prendere qualche carta presente in tavola.

Prima di tutto le carte contenute nella variabile VT (**n**) vengono trasferite in una variabile fittizia **V V** (**n**). Quindi vengono ordinate da quella con valore maggiore a quella con valore minore senza interferire con VT (n) e settando un'altra variabile fittizia P (n) che si occupa di ricordare l'effettiva posizione delle carte riordinate sul tavolo.

Infine, per controllare se c'è la possibilità di prendere, si agisce per sottrazioni successive: dal valore della carta prescelta vengono via via detratti i valori delle carte riordinate: tramite questo meccanismo, se esiste effettivamente una possibilità di prendere, ad un certo punto delle sottrazioni si arriverà al valore **0**.

Se questo avviene, la variabile **X** viene settata a **2**, altrimenti rimane nulla.

E sarà proprio questa variabile ad essere analizzata dalle varie routines.

Per sapere quali sono le carte in tavola che la carta può prendere, le posizioni delle carte stesse vengo registrate nella variabile **T (n)**, mentre **T** conterrà il numero di carte che si possono prendere.

Come sempre, sta a voi cercare di incrementare il livello di difficoltà del gioco inserendo nuove routines di analisi delle scelte del computer, cercando di eliminare del tutto la subroutine *Caso*.

Si potrebbe, ad esempio, fare in modo che l'elaboratore cerchi in ogni sua mossa di prendere, soprattutto se in tavola ci sono carte importanti (la ruotine per prendere i 7 è già presente, ma si potrebbero analizzare anche i *danari*, i *sei* e gli *assi*), stando però attenti a non favorire troppo l'avversario, evitando magari di lasciare facili scope!

```
1 REM RIGHE!
5 REM BY DARIO PISTELLA
10 INPUT"LINEE ";A$:A=VAL(A$)
15 IFA<1ORA>9THENRUN
20 CLS
25 FORY=1TOA:PA(Y)=INT(10*RND(0))+1
27 TT=TT+PA(Y)
30 PRINTY"- ";STRING$(PA(Y),42);
32 PRINT TAB(18)PA(Y)
35 NEXT:TL=A
40 REM *        GIOCATORE        *
45 GI=1:LOCATE15,1:PRINTSPACE$(30)
50 LOCATE15,1:INPUT"LINEA, PALLINE";B,C
55 IFB<1ORB>ATHEN45
60 IFC<1ORC>PA(B)THEN45
65 PA(B)=PA(B)-C:GOSUB185
70 REM *         COMPUTER        *
75 GI=2:B=0:D=0:U=0
80 FORY=1TOA
85 IFPA(Y)>2THENB=Y
90 IFPA(Y)=2THEND=Y
95 IFPA(Y)=1THENU=Y
100 NEXT:IFBANDTL>2THEN140
105 IFTL=1THENB=B+U+D:C=PA(B)-1:GOTO160
110 GOSUB210:C=1:IFB=0THENB=D
115 IFTL=2ANDU>0ANDB>0THENC=PA(B):GOTO160
120 IFTL=2ANDB>0ANDB<>DTHENC=PA(B)-2
125 IFB=0OR(B>0ANDTK=TAANDTL<>2)THENB=U
130 GOTO160
140 REM *        SORTEGGIO       *
145 C=INT((PA(B)-1)*RND(0))+1:GOSUB210
150 IFTA=TKANDCI=0THEN140
155 IFTA<>TKANDCI=1THEN140
160 REM *          STAMPA        *
165 PA(B)=PA(B)-C
170 LOCATE14,1:PRINT"COMPUTER:"
172 PRINTTAB(15)"  "B","C
175 FORK=1TO2000:NEXT:GOSUB185
177 LOCATE14,1:PRINTSPACE$(10)
180 GOTO40
185 REM *         CANCELLA       *
190 LOCATEB,6:PRINTSTRING$(PA(B),42);
192 PRINT SPACE$(10-PA(B));TAB(18)PA(B)
195 IFPA(B)=0THENTL=TL-1
200 TT=TT-C:IFTT=1THEN240
205 RETURN
210 REM *          CHECK         *
215 CI=0:TA=0:TK=0
220 IFTL/2=INT(TL/2)THENTA=1
225 IFTT/2=INT(TT/2)THENTK=1
230 IFC/2=INT(C/2)THENCI=1
235 RETURN
240 REM *           FINE         *
245 CLS:PRINT"VINCITORE=";
247 IFGI=1THENPRINT"TU":ELSE PRINT"IO"
250 END
```

```
1 REM SCOPA GW-BASIC
5 REM BY DARIO PISTELLA
10 DIMCR(2,40),SR(2,40)
15 A$(1)="D":A$(2)="B":A$(3)="S":A$(4)="C":VT=4:CLS
20 REM * DOCARTE *
25 IFCA=40THEN440
30 FORCO=1TO3:FORG=1TO2:GOSUB425
35 IFG=1THENLOCATECO,1:PRINTCO"- "VL;A$(SG)
40 V(G,CO)=VL:S(G,CO)=SG
45 NEXTG,CO:CA=CA+6:IFCA=6THENCA=10
50 LOCATE19,1:PRINT"RIMASTE:";40-CA
55 IFCA>10THENRETURN
60 FORCO=1TO4:GOSUB425
65 LOCATE15,(5*(CO-1))+1:PRINTCO:PRINTTAB(5*(CO-1));VL;A$(SG)
70 VT(CO)=VL:ST(CO)=SG:NEXT
75 REM * ATTESA *
80 GC=1:NV=0:X=0:VA=0:A$=INKEY$
85 A=VAL(A$):IFA<1ORA>3ORV(1,A)=0THEN75
90 LOCATE5,1:PRINTV(1,A);A$(S(1,A))
95 PRINTSPACE$(20):LOCATE6,1:INPUT"CARTE ";A$
100 IFA$="L"THENGOSUB320:GOTO215
105 FORCO=1TOVT:VZ=VAL(MID$(A$,CO,1)):IFVZ<=VTTHENVA=VA+VT(VZ)
110 IFVT(CO)=V(1,A)ANDLEN(A$)>1THENX=1
115 NEXT
120 IF(VA<>V(1,A))ORXTHENLOCATE6,1:PRINT"NON VALIDO":NV=1:GOTO215
125 REM * GIOCATORE PRENDE *
130 CX=0:X=0:LOCATE5,1:PRINTV(GC,A);A$(S(GC,A)):PRINT"CARTE:"A$"     "
135 FORCO=1TO2000:NEXT:LOCATE15,1:PRINTSPACE$(79):PRINTSPACE$(79)
140 FORCO=1TOVT:Y=0
145 FORC2=1TOLEN(A$):CN=VAL(MID$(A$,C2,1))
150 IFCN=COTHENY=1:VC=VT(CO):SC=ST(CO):GOSUB195
155 NEXT:IFYTHEN170
160 CX=CX+1:LOCATE15,(5*(CX-1))+1:PRINTCX:PRINTTAB(5*(CX-1));VT(CO);A$(ST(CO))
165 VT(CX)=VT(CO):ST(CX)=ST(CO)
170 REM * NOPRINT *
175 NEXT:VT=VT-LEN(A$):VC=V(GC,A):SC=S(GC,A):GOSUB195
180 IFVT=0THENPU(GC)=PU(GC)+1:SC(GC)=SC(GC)+1
185 IFGC=1THENGOSUB410
190 UP=GC:GOTO215
195 REM * VALUTAZIONE CARTA *
200 IFSC=1ANDVC=7THENPU(GC)=PU(GC)+1:SQ=GC
205 CT(GC)=CT(GC)+1:CR(GC,CT(GC))=VC:SR(GC,CT(GC))=SC
210 RETURN
215 REM * ANCORA *
220 IFNV=0THENV(GC,A)=0:S(GC,A)=0
225 FORU=1TO1000:NEXT:LOCATE5,1:PRINTSPACE$(20):PRINTSPACE$(20)
230 IFV(1,1)+V(2,1)+V(1,2)+V(2,2)+V(1,3)+V(2,3)=0THENGOSUB20
235 IFNVORGC=2THEN75
240 REM * COMPUTER *
245 SM=0:FORSU=1TOVT:SM=SM+VT(SU):NEXT
250 A$="":GC=2:SY=0:SB=0:SP=0:REM CONTR 7,7D,SCOPA
255 FORSE=1TO3
260 IFV(2,SE)=7THENSY=SE:IFS(2,SE)=1THENSB=SE
265 IFV(2,SE)=SMANDSM<>0THENSP=SE
270 NEXT:IFSPTHENSY=SP
275 IFSY=0THEN300
280 A=SY:GOSUB320:IFX<>2THEN300
```

```
285 X=0:IFSBAND(V(2,SP)=7ORSP=0)THENA=SB
290 REM * COMPUTER PRENDE *
295 FORSE=1TOT:A$=A$+RIGHT$(STR$(T(SE)),1):NEXT:GOTO125
300 REM * CASO *
305 FORSE=1TO3:IFV(2,SE)<>0THENA=SE
310 NEXT:GOSUB320:IFX=2THEN290
315 GOSUB395:GOTO215
320 REM * CALA *
325 X=0:VV(1)=0:FORCO=1TOVT:VV(CO)=VT(CO):P(CO)=CO:NEXT
330 REM * ORDINAMENTO *
335 FORCO=1TOVT-1:IFVV(CO)>=VV(CO+1)THEN350
340 H=VV(CO+1):VV(CO+1)=VV(CO):VV(CO)=H
345 H=P(CO+1):P(CO+1)=P(CO):P(CO)=H:GOTO330
350 REM * OK *
355 NEXT
360 FORCO=1TOVT:V=V(GC,A):IFX<>2THENT=0
365 FORU=COTOVT
370 IFX<>2AND(V-VV(U))>-1THENV=V-VV(U):T=T+1:T(T)=P(U)
375 IFV=0THENX=2
380 NEXTU,CO
385 IFGC=2THENRETURN
390 IFX=2THENLOCATE6,1:NV=1:PRINT"NON VALIDO":RETURN
395 REM * PRINT *
400 VT=VT+1:VT(VT)=V(GC,A):ST(VT)=S(GC,A):LOCATE15,(5*(VT-1))+1
405 PRINTVT:PRINTTAB(5*(VT-1));V(GC,A);A$(ST(VT))
410 REM * CANCELLAZIONE *
415 IFGC=1THENLOCATEA,5:PRINTSPACE$(6)
420 RETURN
425 REM * VALORE-SEGNO *
430 VL=INT(10*RND(0))+1:SG=INT(4*RND(0))+1:IFCK(VL,SG)=1THEN425
435 CK(VL,SG)=1:RETURN
440 REM * FINE *
445 GC=UP:FORCO=1TOVT:VC=VT(CO):SC=ST(CO):GOSUB195:NEXT
450 CLS:PRINTTAB(14)"GIOCATORE";TAB(30)"COMPUTER"
455 PRINT"SCOPE";TAB(18);SC(1);TAB(34);SC(2)
460 PRINT"SETTEBELLO";TAB(16*(SQ-1)+19)"*"
465 PRINT"CARTE";TAB(18);CT(1);TAB(34);CT(2)
470 IFCT(1)>20THENPU(1)=PU(1)+1:ELSE IFCT(1)<20THENPU(2)=PU(2)+1
475 FORGI=1TO2
480 FORCO=1TO4:VX=0:MX(GI,CO)=0:FORC2=1TOCT(GI)
485 IFSR(GI,C2)<>COTHEN515
490 VX=CR(GI,C2)+10:IFVX>17THENVX=10
495 IFCR(GI,C2)=7ORCR(GI,C2)=6THENVX=CR(GI,C2)*3
500 IFCR(GI,C2)=1THENVX=16
505 IFVX>MX(GI,CO)THENMX(GI,CO)=VX
510 IFCO=1ANDSR(1,C2)=1THENDN(1)=DN(1)+1
515 REM * SEGNO NON IN ANALISI *
520 NEXT C2,CO,GI
525 FORCO=1TO2:FORC2=1TO4
530 P(CO)=P(CO)+MX(CO,C2):NEXTC2,CO
535 TB=29:IFP(1)>P(2)THENPU(1)=PU(1)+1:TB=19
540 IFP(2)>P(1)THENPU(2)=PU(2)+1:TB=35
545 PRINT"DANARI:";TAB(18);DN(1);TAB(34);10-DN(1)
550 PRINT"PRIMIERA:";TAB(TB)"*":PRINT
555 IFDN(1)>5THENPU(1)=PU(1)+1
560 IFDN(1)<5THENPU(2)=PU(2)+1
565 PRINT"PUNTI:";TAB(18);PU(1);TAB(34);PU(2)
570 END
```

Facciamo la radiografia al nostro PC

< di Giancarlo Mariani>

< Enciclopedia di routine in Basic, C, Pascal per sistemi Ms - Dos >

Sicuramente tutti avranno avuto modo di utilizzare i numerosi programmi per PC, tipo **Sysid**, **PcTools** o altri, che mostrano la configurazione del proprio computer, ossia la memoria presente, il numero di dischi, le porte seriali, parallele, eccetera. Questi programmi sono molto interessanti dato che fanno capire su che macchina si sta lavorando; ma da dove prendono le informazioni che visualizzano? Nella memoria del PC vi sono molte aree di memoria riservate. Una di queste contiene, appunto, i dati di sistema. Inizia dal segmento **40** esadecimale (**64** dec.), byte **0**, e corrisponde alla tabella **1** che vedete pubblicata.

Inoltre esiste anche un'altra area, a partire dal segmento **F000**, che identifica alcuni parametri del PC e che riportiamo in tabella **2**.

```
CONFIGURAZIONE SISTEMA. by MARIANI G.
=========================================================

NUMERO DI PORTE SERIALI PRESENTI   = 2
  COM1: 03F8
  COM2: 02F8

NUMERO DI PORTE PARALLELE PRESENTI = 1
  LPT1: 0378

MEMORIA DI BASE                  = 640 Kbytes
MODO VIDEO CORRENTE              = 03    (80 x 25 COLOR CGA)
NUMERO DI LETTORI FLOPPY DISK    = 2
NUMERO DI HARD DISK PRESENTI     = 1
COPROCESSORE MATEMATICO          = Non Presente
```

Esempio di schermata visualizzabile con i programmi di queste pagine.

Naturalmente la mappa di memoria descritta nel presente articolo non è completa, dal momento che esistono moltissime altre locazioni contenenti le funzioni più disparate.

Nelle routine che vedete in queste pagine proponiamo un programma, implementato in **C**, **Pascal** e **Basic**, che visualizza alcuni dati sulla configurazione di sistema, come ad esempio quella riportata nell'apposito riquadro che, ovviamente, raffigura dati variabili a seconda della configurazione posseduta.

Il funzionamento del programma è semplicissimo perché si limita a leggere le aree di memoria interessate ed a visualizzarne i valori opportunamente trattati.

Una descrizione completa del programma sarebbe pertanto inutile, e quindi ci limitiamo a descrivere le istruzioni usate nelle tre implementazioni.

QuickBasic

L'istruzione per leggere una locazione di memoria in QuickBasic è **Peek**, di commodoriana memoria.

La sintassi è...

`Valore = PEEK (Indirizzo)`

...in cui **Indirizzo** è l'indirizzo che contiene il **Valore** da leggere.

```
40:00   => Indirizzo COM1
40:02   => Indirizzo COM2
40:04   => Indirizzo COM3
40:06   => Indirizzo COM4
40:08   => Indirizzo LPT1
40:0A   => Indirizzo LPT2
40:0C   => Indirizzo LPT3
40:0E   => Indirizzo LPT4 (non valido per i PS/2)
40:10   => Configurazione generale:
  bit 0 => 0 = floppy drive non presente
           1 = floppy drive presente (vedere bit 6,7)
  bit 1 => 0 = Coprocessore matematico non installato
           1 = Coprocessore matematico 80x87 installato
  bit 2,3 => Ram di sistema
             00 = 16k, 01 = 32k, 10 = 48k, 11 = 64k
  bit 4,5 => Modo video iniziale
             00 = nessun adattatore video
             01 = colore 40 colonne
             10 = colore 80 colonne
             11 = MDA
  bit 6,7 => Numero di floppy disk drives
             00 = 1, 01 = 2, 10 = 3, 11 = 4
  bit 8 => 0 = DMA presente
           1 = DMA non presente
  bit 9,A,B => Numero di porte seriali presenti
  bit C => 0 = Adattatore giochi non presente
           1 = Adattatore giochi presente
  bit E,F => Numero di porte parallele installate
40:13   => Ampiezza memoria di sistema in Kbytes
40:17   => Flag per tastiera byte 0:
   bit 7 = modo inserimento on, 6 = capslock on
   bit 5 = numlock on         , 4 = scroll lock on
   bit 3 = 'alt' premuto      , 2 = 'ctrl' premuto
   bit 1 = 'Shift sx' premuto , 0 = 'shift dx' premuto
40:18   => Flag per tastiera byte 1:
   bit 7 = 'insert' premuto   , 6 = 'capslock' premuto
   bit 5 = 'numlock' premuto  , 4 = 'scrol lock' premuto
   bit 3 = tasto 'pause'
40:49   => Modo video corrente
   00h = 40x25 BW, 01h = 40x25 Color, 02h = 80x25 BW
   03h = 80x25 Color,04h = 320x200 Color,05h = 320x200 BW
   06h = 640x200 BW, 07h = Monocromatico
   08h = bassa ris. 16 colori, 09h = med.risol. 16 colori
   0Ah = alta ris. 4 colori,   0Ch = med.risol. 16 colori
   0Dh = alta ris. 16 colori,  0Eh = alta ris. 4 colori
   0Fh = alta ris. 64 colori.
40:4A   => Numero di colonne dello schermo
40:75   => Numero di hard-disk installati
40:78   => (4 bytes) Timeout per LPT1,LPT2,LPT3,LPT4
40:7C   => (4 bytes) Timeout per COM1,COM2,COM3,COM4
```

Area di memoria per i dati di sistema

Dal momento che la memoria del PC è organizzata in **segmenti**, non è sufficiente specificare un indirizzo ma occorrerà far sapere il segmento nel quale è posto quel particolare indirizzo.

Occorre quindi attivare un'altra istruzione Basic, **Def Seg**, che si usa nel modo seguente:

```
DEF SEG = Indirizzo Segmento
```

...in cui **Indirizzo Segmento** è l'indirizzo iniziale del **segmento** desiderato.

Per determinare il numero di hard disk presenti su un PC si deve attivare, ad esempio, il seguente programmino:

```
DEF SEG = &h40 :
REM Seleziona segmento di sistema
Nhard = PEEK (40:&h75)
REM Legge il dato
DEF SEG :
REM Ripristina il segmento originale
PRINT "Numero di hard-disk= ";Nhard
```

L'istruzione Def Seg, priva di parametri, serve per ripristinare il segmento di default del Basic, ossia quello nel quale il Basic lavora, alloca le variabili, e così via. E' molto importante che sia presente tale istruzione, altrimenti il Basic resta posizionato sul segmento video e possono accadere cose imprevedibili.

Per la stessa ragione l'istruzione Def Seg andrebbe sempre scritta in un programma e **mai** impartita in modo diretto. Inoltre deve essere nota la funzione svolta dalle varie locazioni specificate tramite l'indirizzo in Def Seg e nelle eventuali istruzioni Peek / Poke.

Turbo-C

Il C possiede, al suo interno, un'istruzione che esegue sia il cambio di segmento che la lettura del byte o della word. Le istruzioni in questione sono **Peekb** e **Peek**; si usano così:

`Byte = peekb` (Segmento, Indirizzo); per leggere 1 byte.

`Word = peek` (Segmento, Indirizzo); per leggere 1 word.

In caso di lettura word, il valore sarà determinato dal contenuto dell'indirizzo specificato, più il contenuto del successivo moltiplicato per 256.

Segmento è l'indirizzo iniziale del segmento da prendere in considerazione, **Indirizzo** è l'indirizzo relativo al segmento dal quale va letto il valore. **Byte** e **Word** sono i valori ottenuti.

```
F000:FFF5 => Data della release del BIOS
F000:FFFE => Identificazione del modello di PC:
```

modello	sottomod	revisione	macchina
FA	00	00	PS/2 mod. 30
FB	00	01	XT
FB	01		XT-2
FB	01		XT-2 (revisione)
FC			AT
FC	00	01	AT mod.239
FC	01	00	AT mod.319
FC	02	00	XT/286
FC	04	00	PS/2 mod. 50
FC	05	00	PS/2 mod. 60
FC	00		7531/2 ind. AT
FD			PCjr
FE			XT portatile
2D			Compaq PC
9A			Compaq Plus

Tabella 2 su vari parametri del PC

Lo stesso programma, visto prima, scritto in C diventa:

```
Nhard = peekb(0x0040, 0x75);/* Legge
il dato*/
printf("Numero di hard disk =
%d\n",Nhard);
```

Turbo Pascal

Il Turbo Pascal si differenzia un po' dagli altri due linguaggi dal momento che **non** possiede istruzioni specifiche per scrivere in memoria. Il Pascal, però, possiede, già definito al suo interno, un **array** di interi chiamato **Mem** che contiene tanti elementi quanti sono i bytes della memoria del PC.

Ognuno degli elementi è l'immagine del corrispondente byte di memoria. Per leggere un byte o una word dalla memoria del PC in Pascal dovremo quindi fare:

```
Byte = Mem [Segmento:Indirizzo]
        ;legge un byte
Word =Memw [Segmento:Indirizzo]
        ;legge una word
```

Anche in questo caso, se si legge una word, il valore recuperato sarà dato dal contenuto di *segmento:indirizzo* sommato al contenuto di segmento:indirizzo + 1 moltiplicato per 256.

Segmento è l'indirizzo iniziale del segmento da prendere in considerazione, **Indirizzo** è l'indirizzo relativo al segmento dal quale va letto il valore. Byte e Word sono i valori recuperati.

Il programma visto prima, scritto in Pascal, quindi diventa:

```
Nhard := Mem [64:117]
        ;{ Legge il dato }
writeln('Numero di hard disk =
',Nhard);
```

Le parti comuni

Nei tre listati, la routine **PrintHex** esegue una conversione di **Valore** da decimale a esadecimale per scrivere il risultato usando il numero di cifre specificato da **c**.

Nel caso del Basic, la conversione viene effettuata automaticamente dall'istruzione HEX$...

```
Stringa$ = HEX$ (Valore)
```

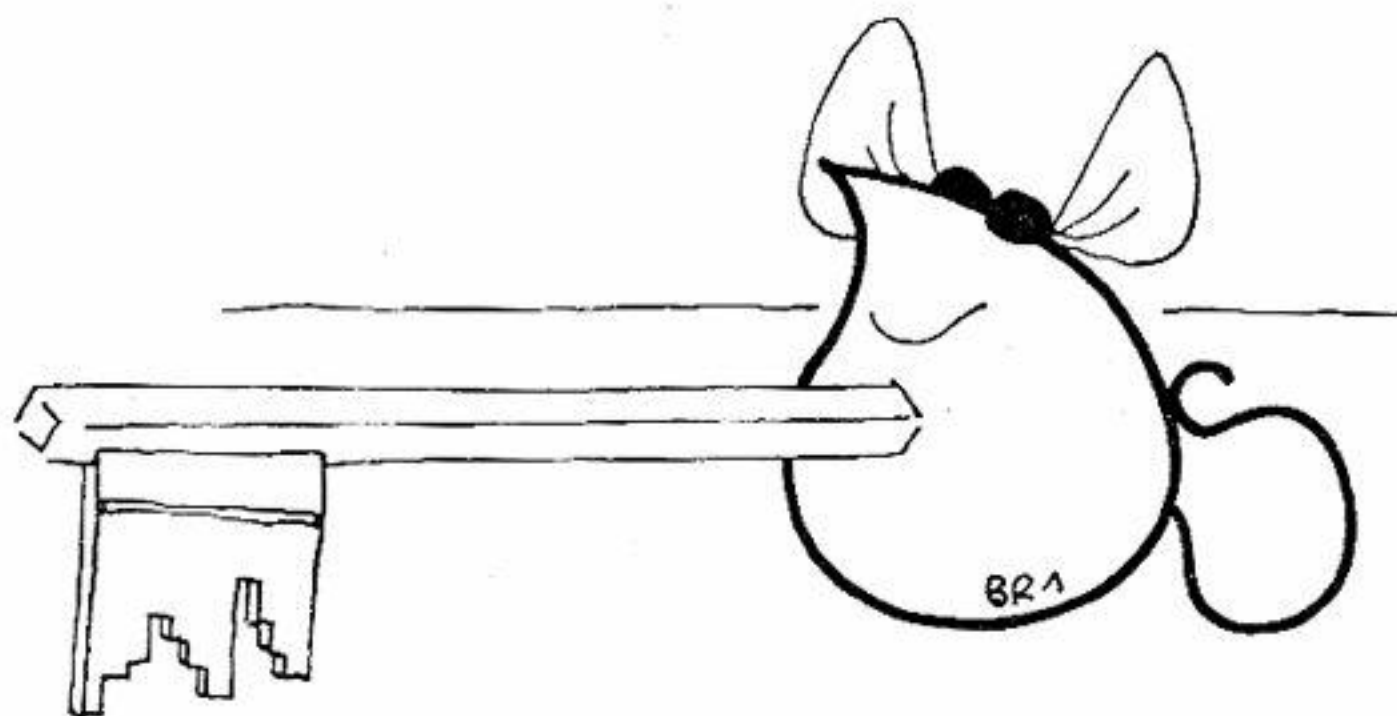
...dove **Valore** è il valore da convertire, mentre **Stringa$** è la variabile stringa nella quale viene inserito il risultato.

Nel caso del C, la conversione viene effettuata dall'istruzione **ITOA**...

```
ITOA (Valore, Radice, Stringa);
```

...che converte un numero intero in una stringa.

Il numero è contenuto nella variabile **Valore**; **Radice** indica in che base numerica deve essere effettuata la conversione (nel nostro caso, **16** = esadecimale); **Stringa** è il puntatore alla stringa nella quale deve essere depositato il risultato. La funzione stessa ITOA restituisce il puntatore alla stringa risultato.

In Pascal, invece, la conversione viene eseguita manualmente con i metodi tradizionali (scomposizione in parte alta / bassa, ecc.).

```c
/*                                                                */
/* CONFIG.C: Scrive la configurazione di sistema. Versione Turbo-C */
/*                                                                */
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <conio.h>
#include <string.h>

void PrintHex(long Valore, int c);
void RecuperaDati(void);

/* Dichiarazione variabili */

  long ComX[4];          /* Variabili per indirizzo COMx   */
  long LptX[4];          /* Variabili per indirizzo LPTx   */
  long BaseMem;          /* Memoria di base                */
  long ModoVideo;        /* Modo video corrente            */
  long Num;              /* Variabile generale             */
  long Config;           /* Configurazione generale        */
  unsigned char a[50];   /* Stringa generale               */
  int Nhard;             /* Numero di hard-disk            */
  int k;                 /* Variabile x cicli FOR          */

/* Converte il valore "Valore" in esad., e lo scrive lungo "c" cifre */
void PrintHex (long Valore, int c)
{
  unsigned char Dummy[30],Stringa[30];
  strcpy(Stringa,"00000000");
  strcat(Stringa,itoa(Valore,Dummy,16)); strupr(Stringa);
  printf("%s",Stringa+strlen(Stringa)-c);
}

/* Recupera i dati di configurazione dalla memoria */
void RecuperaDati()
{
  int k,Addr;

/* Recupera indirizzi seriale COM */
  for (k=1; k<=4; k++)
  {
    Addr = (k - 1) * 2;
    ComX[k-1] = peek(64,Addr);
  }

/* Recupera indirizzi interfacce parallele LPT */
  for (k=1; k<=4; k++)
  {
    Addr = ((k - 1) * 2) + 8;
    LptX[k-1] = peek(64,Addr);
  }

/* Recupera memoria di base */
  BaseMem = peek(64,19);

/* Recupera modo video corrente */
  ModoVideo = peekb(64,73);

/* Recupera numero di hard disk attaccati */
  Nhard = peekb(64,117);

/* Recupera configurazione generale */
  Config = peek(64,16);

}

void main()
{
  clrscr(); printf("Attendere... sto leggendo i dati\n");
  RecuperaDati();
  clrscr();
  printf("CONFIGURAZIONE SISTEMA. by MARIANI G.\n");
  printf("==================================================\n\n\n");

  Num = (Config & 3584) / 512;
  printf("NUMERO DI PORTE SERIALI PRESENTI    = %d\n", Num);

/* Scrive indirizzi seriale COM */
  for (k=0; k<4; k++)
  {
    if (ComX[k] != 0)
    {
      printf("  COM%d: ",k+1);
      PrintHex(ComX[k], 4); printf("\n");
    }
  }
  printf("\n");

  Num = (Config & 49152) / 16384;
  printf("NUMERO DI PORTE PARALLELE PRESENTI = %d\n",Num);

/* Scrive indirizzi porte parallele LPT */
  for (k=0; k<4; k++)
  {
    if (LptX[k] != 0)
    {
      printf("  LPT%d: ",k+1);
      PrintHex(LptX[k], 4); printf("\n");
    }
  }
  printf("\n");

/* Scrive memoria di base */
  printf("MEMORIA DI BASE                    = %d Kbytes\n",BaseMem);

/* Scrive modo video */
  printf("MODO VIDEO CORRENTE                = ");
  PrintHex(ModoVideo, 2);
  switch (ModoVideo)
  {
    case 0: { strcpy(a,"40 x 25 BW - CGA"); break; }
    case 1: { strcpy(a,"40 x 25 COLOR - CGA"); break; }
    case 2: { strcpy(a,"80 x 25 BW - CGA"); break; }
    case 3: { strcpy(a,"80 x 25 COLOR - CGA"); break; }
    case 4: { strcpy(a,"320 x 200 COLOR - CGA"); break; }
    case 5: { strcpy(a,"320 x 200 BW - CGA"); break; }
    case 6: { strcpy(a,"640 x 200 BW - CGA"); break; }
    case 7: { strcpy(a,"MONOCHROME - MDA"); break; }
    default: { strcpy(a,"SCONOSCIUTO"); break; }
  }
  printf("   (%s)\n",a);

/* Scrive numero di floppy disk attaccati */
  Num = ((Config & 192) / 64) + 1;
  printf("NUMERO DI LETTORI FLOPPY DISK      = %d\n", Num);

/* Scrive numero di hard disk attaccati */
  printf("NUMERO DI HARD DISK PRESENTI       = %d\n", Nhard);

/* Scrive se processore matematico presente oppure no */
  printf("COPROCESSORE MATEMATICO            = ");
  if ((Config & 2) == 0)
    printf("Non Presente\n");
  else
    printf("Presente (80x87)\n");

}
```

La versione C dell'algoritmo. Nei prossimi due riquadri è pubblicata la versione Turbo Pascal

```
{                                                                      }
{ CONFIG.PAS: Scrive la configurazione di sistema. Versione Turbo-Pascal}
{                                                                      }
program Configurazione;

uses Crt;

procedure PrintHex(Valore:longint; c:integer); forward;
procedure RecuperaDati; forward;

{ Dichiarazione variabili }

var
  ComX:array[1..4] of longint;    { Variabili per indirizzo COMx }
  LptX:array[1..4] of longint;    { Variabili per indirizzo LPTx }
  BaseMem:longint;                { Memoria di base              }
  ModoVideo:longint;              { Modo video corrente          }
  Num:longint;                    { Variabile generale           }
  Config:longint;                 { Configurazione generale      }
  a:string;                       { Stringa generale             }
  Nhard:integer;                  { Numero di hard-disk          }
  k:integer;                      { Variabile x cicli FOR        }

{ Converte il valore "Valore" in esad. e lo scrive lungo "c" cifre }
procedure PrintHex (Valore:longint; c:integer);
var
  s,s2:string;
  High,Low:integer;
  H1,L1,H2,L2:integer;
begin
  s:='0123456789ABCDEF';
  High:=Hi(Valore);
  Low:=Lo(Valore);
  H1:=High div 16; L1:=High-(H1*16);
  H2:=Low div 16; L2:=Low-(H2*16);
  s2:='00000000'+copy(s,H1+1,1)+copy(s,L1+1,1);
  s2:=s2+copy(s,H2+1,1)+copy(s,L2+1,1);
  write(copy(s2,13-c,c));
end;

{ Recupera i dati di configurazione dalla memoria }

procedure RecuperaDati;
var
  k,Addr:integer;
begin

{ Recupera indirizzi seriale COM }
  for k := 1 to 4 do
  begin
    Addr := (k - 1) * 2;
    ComX[k] := memw[64:Addr];
  end;

{ Recupera indirizzi interfacce parallele LPT }
  for k := 1 to 4 do
  begin
    Addr := ((k - 1) * 2) + 8;
    LptX[k] := memw[64:Addr];
  end;

{ Recupera memoria di base }
  BaseMem := memw[64:19];

{ Recupera modo video corrente }
  ModoVideo := mem[64:73];

{ Recupera numero di hard disk attaccati }
  Nhard := mem[64:117];

{ Recupera configurazione generale }
  Config := memw[64:16];
end;

begin { Main Program }
  clrscr; writeln('Attendere... sto leggendo i dati');
  RecuperaDati;
  clrscr;
  writeln('CONFIGURAZIONE SISTEMA. by MARIANI G.');
  writeln('=========================================================');
  writeln; writeln;

  Num := (Config and 3584) div 512;
  writeln('NUMERO DI PORTE SERIALI PRESENTI   = ', Num);

{ Scrive indirizzi seriale COM }
  for k:= 1 to 4 do
  begin
    if (ComX[k] <> 0) then
    begin
      write('  COM',k,': ');
      PrintHex(ComX[k], 4); writeln;
    end;
  end; writeln;

  Num := (Config and 49152) div 16384;
  writeln('NUMERO DI PORTE PARALLELE PRESENTI = ',Num);

{ Scrive indirizzi porte parallele LPT }
  for k:= 1 to 4 do
  begin
    if (LptX[k] <> 0) then
    begin
      write('  LPT',k,': ');
      PrintHex(LptX[k], 4); writeln;
    end;
  end; writeln;

{ Scrive memoria di base }
  writeln('MEMORIA DI BASE                    = ',BaseMem,' Kbytes');

{ Scrive modo video }
  write('MODO VIDEO CORRENTE                = ');
  PrintHex(ModoVideo, 2);
  case ModoVideo of
    0: a:= '40 x 25 BW - CGA';
    1: a:= '40 x 25 COLOR - CGA';
    2: a:= '80 x 25 BW - CGA';
    3: a:= '80 x 25 COLOR - CGA';
    4: a:= '320 x 200 COLOR - CGA';
    5: a:= '320 x 200 BW - CGA';
    6: a:= '640 x 200 BW - CGA';
    7: a:= 'MONOCHROME - MDA';
  else
    a:='SCONOSCIUTO';
  end;
  writeln('   (',a,')');

{ Scrive numero di floppy disk attaccati }
  Num := ((Config and 192) div 64) + 1;
  writeln('NUMERO DI LETTORI FLOPPY DISK      = ', Num);

{ Scrive numero di hard disk attaccati }
  writeln('NUMERO DI HARD DISK PRESENTI       = ', Nhard);

{ Scrive se processore matematico presente oppure no }
  write('COPROCESSORE MATEMATICO            = ');
  if (Config and 2) = 0 then
    writeln('Non Presente')
  else
    writeln('Presente (80x87)');

end.
```

La versione Quick Basic Microsoft

```
'
' CONFIG.BAS: Scrive la configurazione di sistema. Versione QuickBasic
'
DECLARE SUB PrintHex (Valore AS LONG, c AS INTEGER)
DEFINT A-Z

'Dichiarazione variabili

  DIM ComX(1 TO 4)  AS LONG      'Variabili per indirizzo COMx
  DIM LptX(1 TO 4)  AS LONG      'Variabili per indirizzo LPTx
  DIM BaseMem AS LONG            'Memoria di base
  DIM ModoVideo AS LONG          'Modo video corrente
  DIM Num AS LONG                'Variabile generale
  DIM Config AS LONG             'Configurazione generale

  CLS : PRINT "Attendere... sto leggendo i dati"
  GOSUB RecuperaDati
  CLS
  PRINT "CONFIGURAZIONE SISTEMA. by MARIANI G."
  PRINT STRING$(79, "="): PRINT : PRINT

  Num = (Config AND &HE00) / 512
  PRINT "NUMERO DI PORTE SERIALI PRESENTI    = "; Num

'Scrive indirizzi seriale COM
  FOR k = 1 TO 4
    IF ComX(k) <> 0 THEN
      PRINT "  COM"; STR$(k); ": ";
      CALL PrintHex(ComX(k), 4): PRINT
    END IF
  NEXT k: PRINT

  Num = (Config AND &HC000) / 16384
  PRINT "NUMERO DI PORTE PARALLELE PRESENTI = "; Num

'Scrive indirizzi parallele LPT
  FOR k = 1 TO 4
    IF LptX(k) <> 0 THEN
      PRINT "  LPT"; STR$(k); ": ";
      CALL PrintHex(LptX(k), 4): PRINT
    END IF
  NEXT k: PRINT

'Scrive memoria di base
  PRINT "MEMORIA DI BASE                     = "; BaseMem; " Kbytes"

'Scrive modo video
  PRINT "MODO VIDEO CORRENTE                 = ";
  CALL PrintHex(ModoVideo, 2)
  SELECT CASE ModoVideo
    CASE 0: a$ = "40 x 25 BW - CGA"
    CASE 1: a$ = "40 x 25 COLOR - CGA"
    CASE 2: a$ = "80 x 25 BW - CGA"
    CASE 3: a$ = "80 x 25 COLOR - CGA"
    CASE 4: a$ = "320 x 200 COLOR - CGA"
    CASE 5: a$ = "320 x 200 BW - CGA"
    CASE 6: a$ = "640 x 200 BW - CGA"
    CASE 7: a$ = "MONOCHROME - MDA"
    CASE ELSE: a$ = "SCONOSCIUTO"
  END SELECT
  PRINT "   ("; a$; ")"

'Scrive numero di floppy disk attaccati
  Num = ((Config AND &HC0) / 64) + 1
  PRINT "NUMERO DI LETTORI FLOPPY DISK       = "; Num

'Scrive numero di hard disk attaccati
  PRINT "NUMERO DI HARD DISK PRESENTI        = "; Nhard

'Scrive se processore matematico presente oppure no
  PRINT "COPROCESSORE MATEMATICO             = ";
  IF (Config AND 2) = 0 THEN
    PRINT "Non Presente"
  ELSE
    PRINT "Presente (80x87)"
  END IF

END

RecuperaDati:

  DEF SEG = &H40

'Recupera indirizzi seriale COM
  FOR k = 1 TO 4
    Addr = (k - 1) * 2
    ComX(k) = PEEK(Addr) + 256 * PEEK(Addr + 1)
  NEXT k

'Recupera indirizzi interfacce parallele LPT
  FOR k = 1 TO 4
    Addr = ((k - 1) * 2) + 8
    LptX(k) = PEEK(Addr) + 256 * PEEK(Addr + 1)
  NEXT k

'Recupera memoria di base
  BaseMem = PEEK(&H13) + PEEK(&H14) * 256

'Recupera modo video corrente
  ModoVideo = PEEK(&H49)

'Recupera numero di hard disk attaccati
  Nhard = PEEK(&H75)

'Recupera configurazione generale
  Config = PEEK(&H10) + 256 * PEEK(&H11)

DEF SEG
RETURN

'
'Converte il valore "Valore" in esad. e lo scrive lungo "c" cifre
'
SUB PrintHex (Valore AS LONG, c AS INTEGER)

  a$ = HEX$(Valore)
  a$ = STRING$(c, "0") + a$
  a$ = RIGHT$(a$, c)
  PRINT a$;

END SUB
```

Te la do io la directory!

< di Giancarlo Mariani>

< Esaminare C. C. n. 84 >

Anche questo mese continuiamo con la trattazione delle chiamate al DOS (istruzione INT) presentando nuove istruzioni assembler e direttive del compilatore.

Diamo ora una breve descrizione dell'istruzione INT, per chi ancora non la conoscesse.

INT serve per richiamare specifiche routine presenti nel BIOS, nel DOS del PC oppure delle routine costruite dal programmatore per svolgere determinati compiti.

Tale particolare consente di sfruttare, in propri programmi, le routine già costruite e residenti nel PC.

INT si usa con la sintassi...

```
INT Interrupt
```

...in cui INT è il codice mnemonico dell'istruzione, abbreviazione dell'inglese INTerrupt, ossia interruzione; **Interrupt** è il numero dell'interruzione da richiamare, e può essere *solo* un valore immediato.

I vari parametri da passare all'istruzione vengono immessi nei registri (AX, BX, CX, ecc.) *prima* della chiamata.

Le funzioni di cui ci occuperemo in questa puntata sono dedicate alla gestione del disco e riguardano, in particolare, la ricerca dei file. Vediamole assieme:

INT 21h, Funzione 1Ah

Setta l'area di transito (DTA, data transit area) per il disco e serve per settare un'area di memoria che verrà utilizzata dal PC per i dati letti e scritti da disco.

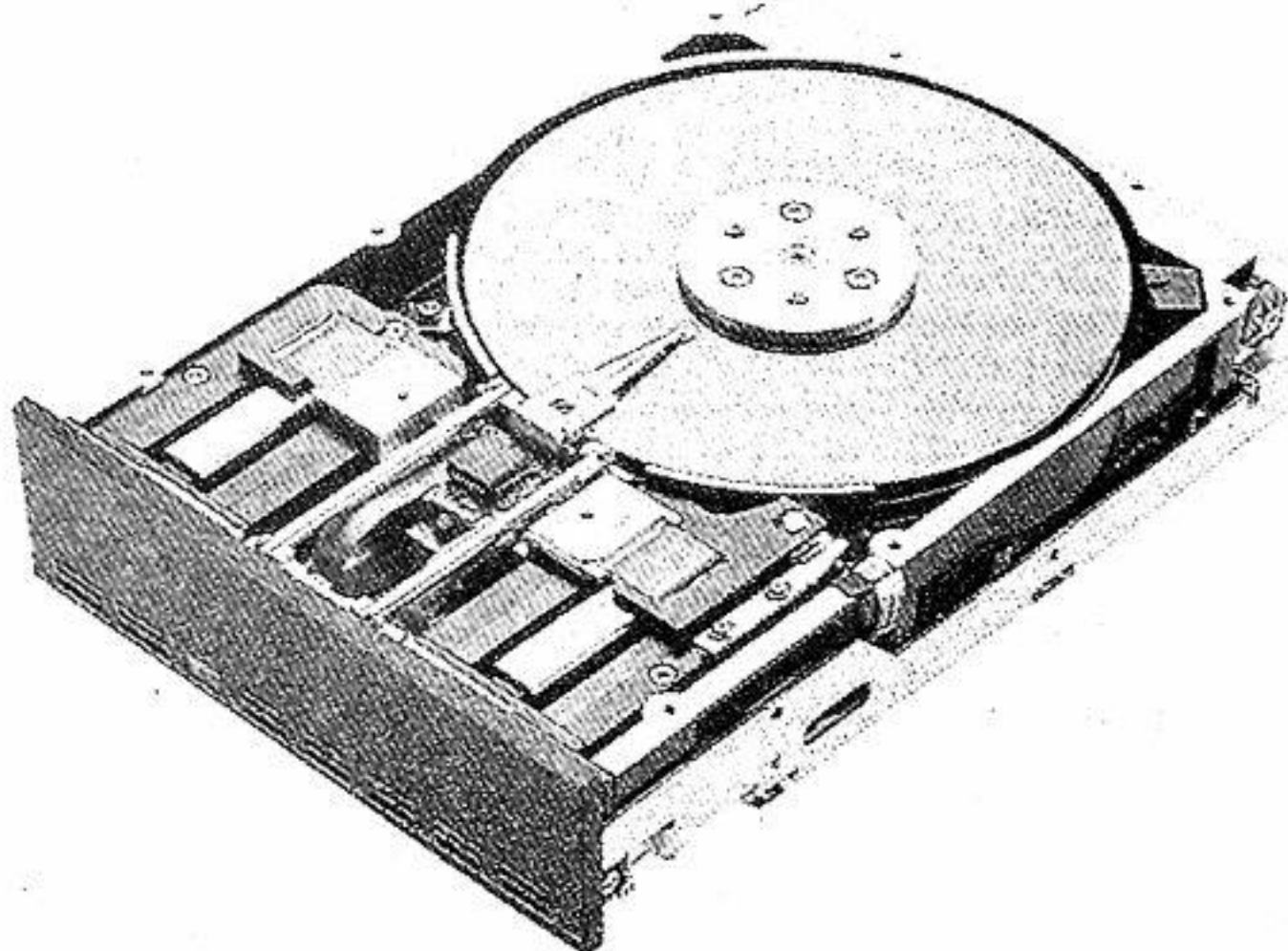
I parametri da passare alla funzione sono:

ah = 1Ah (funzione 1A)

ds:dx = segmento e offset dell'area di memoria da utilizzare, che deve essere ampia un numero sufficiente di byte per contenere le informazioni richieste.

INT 21h, Funzione 29h

Costruisce un file control block (FCB) da un nome file e viene usata per costruire un blocco di controllo file partendo da una semplice stringa, come...

"DRIVE:NomeFile"

MOV OFFSET

MOV è il codice mnemonico (assembler) dell'istruzione, ed è l'abbreviazione della parola inglese "move", ossia muovi. Lo scopo di MOV è infatti quello di muovere, o meglio, copiare, il contenuto di una zona di memoria in un'altra. In questo caso, nel parametro **destinazione** sarà immesso un indirizzo.

L'istruzione MOV era già stata vista nel n. **81** di Computer Club. Questa volta viene presentata in forma leggermente diversa. Il parametro OFFSET, infatti, specifica che **non** è un dato da immettere nel registro **destinazione**, ma un **Indirizzo**.

Sorgente. E' la label della quale vogliamo determinare l'indirizzo.

MOV	Destinazione	OFFSET	Sorgente

Destinazione. E' il registro o la locazione di memoria nella quale andrà a finire il contenuto di ciò che leggiamo. Destinazione può essere:
- Uno dei **registri** : AX, BX, CX, DX (anche nelle forme a 8 bit AH, AL, BH, BL, CH, CL, DH, DL), SI, DI, SP, BP, CS, DS, SS, ES.
- Una qualsiasi **locazione di memoria** (espressa come offset tra 0 e FFFF relativo ad un segmento)
- Un **puntatore** ad una locazione, espresso da un registro o da una coppia di locazioni.

Esempi:

`MOV AX, OFFSET Pippo`

In **AX** viene caricato l'indirizzo corrispondente alla label **Pippo**. (Indirizzo stringa *"Ciao"*)

`MOV DS:DX, OFFSET Pluto`

In DS:DX viene caricato l'indirizzo corrispondente alla label **Pluto**. (Indirizzo stringa *"Salve"*)

```
Pippo: db 'Ciao'
Pluto: db 'Salve'
```

Il blocco di controllo file è utilizzato dal DOS del PC per accedere ai file su disco. I parametri della funzione sono:

ah = 29h (funzione 29)

ds : si = segmento e offset dell'area di memoria contenente il nome del file

es : di = segmento e offset dell'area di memoria nel quale si vuole scrivere il FCB.

al = maschera di controllo. Normalmente posta a zero.

INT 21h, Funzione 11h

Ricerca il primo file specificato da FCB e viene utilizzata per individuare, su disco, il primo file specificato dall'area di memoria FCB. I parametri da passare alla funzione sono:

ah = 11h (funzione 11)

ds : dx = segmento e offset dell'area di memoria contenente il FCB.

La funzione restituisce:

AL = 00h (= file trovato)

AL = FFh (file non trovato)

Nel primo caso le informazioni vengono inserite nell'area di memoria DTA, specificata dalla funzione **1Ah**. Dal byte **1** al byte **11** di tale area sarà presente il nome del file (8 caratteri di nome + 3 di estensione) scritto in formato ASCII.

INT 21h, Funzione 12h

Ricerca il file successivo; la funzione è la continuazione logica della precedente dal momento che ricerca, infatti, il successivo nome di file specificato da FCB; deve essere usata *dopo* la funzione 11h oppure *dopo* un'altra funzione 12h.

In ogni caso, la prima volta che si ricerca un file deve essere usata la funzione 11h.

I parametri da passare alla funzione sono:

ah = 12h (funzione 12)

ds : dx = segmento e offset dell'area di memoria contenente il FCB.

Restituisce...

AL = 00h (file trovato)

AL = FFh (file non trovato)

Nel primo caso le informazioni vengono inserite nell'area di memoria DTA, specificata dalla funzione 1Ah. Dal byte 1 al byte 11 di tale area sarà presente il nome del file (8 caratteri di nome + 3 di estensione) sempre in formato ASCII.

INT 21h, Funzione 4Ch

Termina con codice di ritorno. La funzione permette di terminare un programma e di ritornare al DOS o al programma chiamante. I parametri da passare alla funzione sono i seguenti:

ah = 4Ch (funzione 4C)

al = Codice di ritorno (normalmente 0)

Il codice di ritorno può essere interpretato in procedure BATCH con istruzioni del tipo IF ERRORLEVEL... e quindi può essere fornito per determinare se un programma è andato a buon fine oppure no.

La funzione si usa nel seguente modo:

```
mov ah, 4Ch ; Funzione 4C
mov al, 0   ; Codice di ritorno (Normalmente 0)
int 21h     ; Ritorna al programma chiamante
```

Nell'uso dell'istruzione INT bisogna fare moltissima attenzione sia a richiamare interrupt "esistenti", sia a passare correttamente i parametri a questi ultimi. Se si richiama un interrupt non esistente, oppure si sbaglia il passaggio di parametri, il PC potrebbe inchiodarsi, costringendo quindi a resettare.

Direttive del compilatore

Nel programma presentato in queste pagine viene utilizzata una **nuova**

PUSH e POP

Push, Pop sono i codici mnemonici (assembler) delle istruzioni.

Registro. Specifica che cosa salvare o recuperare dallo stack.

Registro può essere:

- Uno dei **registri** : AX, BX, CX, DX (anche nelle forme a 8 bit AH, AL, BH, BL, CH, CL, DH, DL), SI, DI, SP, BP, CS, DS, SS, ES.
- Un **valore immediato** (solo per Push)

Le istruzioni non influenzano alcun flag.

Esempi:

Push AX ; Salva AX nello stack

Pop BX ; Recupera un elemento dallo stack e lo mette in BX.

Le istruzioni **PUSH** (Spingi) e **POP** (Togli) servono, rispettivamente, a salvare ed a recuperare dei registri nello **Stack**. Lo stack è l'area di memoria dedicata alla memorizzazione degli indirizzi di ritorno e dei parametri delle subroutine. Quest'area è gestita in modo LIFO, (**L**ast-**I**n, **F**irst-**O**ut) ossia l'ultimo elemento ad essere memorizzato è il primo ad essere tirato fuori.

L'istruzione **Push** aggiunge un elemento in cima a quest'area di memoria (immaginate di avere una pila di piatti bianchi e di aggiungerne uno rosso in cima).

L'istruzione **Pop** recupera il primo elemento in cima alla lista (nell'esempio, il piatto rosso).

Attenzione a:

Ad ogni Push *deve* corrispondere un Pop. In pratica lo stack deve essere lasciato come si trovava prima di utilizzare le istruzioni Push / Pop perché lo stack viene utilizzato dal PC per memorizzare gli indirizzi di ritorno delle subroutine e dei programmi. Se lo viene "sporcato" tramite Push / Pop, il computer si inchioda.

Ad esempio, la sequenza...

```
PUSH BX     ; Salva BX nello stack
CALL Pippo
... ... ...
Pippo:
POP BX ; Toglie un elemento dallo stack
... ... ...
RET ; Ritorno (?)
```

...è sbagliatissima perché al momento dell'istruzione **Call**, il PC mette sullo stack gli indirizzi di ritorno dalla procedura "Pippo", quindi il **BX** salvato da Push viene a trovarsi ora non più in cima alla lista, ma *dopo* gli indirizzi. Quindi l'istruzione Pop, posta all'interno della procedura Pippo, **non** recupera il BX salvato, ma recupera un indirizzo di ritorno. La RET, quindi, non troverà più quell'indirizzo nello stack e quasi sicuramente si produrrà un blocco del computer. La sequenza corretta è invece la seguente:

```
PUSH BX ;Salva BX
Call Pippo
POP BX ;Recupera BX
```

Nella subroutine Pippo **non deve** assolutamente essere toccato lo stack tramite istruzioni Push / Pop. Se il computer si inchioda, nonostante tutto, è probabile che all'interno della routine Pippo vi siano alcune istruzioni che manipolano lo stack; morale: usate solo routine di cui conoscete benissimo il funzionamento.

direttiva del compilatore che serve per creare delle aree dati.

Nei fascicoli precedenti abbiamo visto che, tramite le direttive **DB** e **DW**, si creavano aree dati lunghe, rispettivamente, **1** e **2** byte. Per creare un'area più lunga viene sempre usata la direttiva DB, modificata come segue...

```
DB nn DUP (v)
```

...in cui **nn** è il numero di byte da creare e **v** il valore di inizializzazione di quei byte. Ad esempio...

```
Pippo: DB 150 DUP (0)
```

...crea, all'indirizzo **Pippo** un'area di memoria lunga **150** byte, riempita con il valore **0.** Il parametro **v** può anche essere un punto di domanda (**?**). In questo caso l'area di memoria viene creata, ma **non** viene riempita con nessun valore. Resta, cioè, non inizializzata ed il contenuto può esser casuale.

Il programma

Il programma di queste pagine lista su video i nomi dei file contenuti nella directory corrente di un disco (hard o floppy).

In pratica il programma è una notevole semplificazione del comando DIR, e consente di vedere come si può gestire una directory in assembler.

All'inizio troviamo le solite inizializzazioni dei registri di segmento **DS** ed **ES**, spiegate in precedenza.

La successiva chiamata all'interrupt **21h**, funzione **1Ah**, decide in quale area di memoria si vogliono inserire le informazioni lette da disco.

In questo caso l'area di memoria scelta è quella specificata dalla label **dta**, definita nello stesso programma come lunga **50** byte.

Il successivo passo richiama di nuovo l'interrupt **21h**, questa volta con la funzione **29h**.

Tale chiamata non fa altro che prendere il nome del file specificato dall'*indirizzc match*, che nel programma presentato corrisponde alla stringa ***.*** (vedere dichiarazione dati).

La stringa viene trasformata in una stringa interpretabile dalle funzioni DOS di ricerca dei file e quindi messa nell'area di memoria specificata dalla label **fcb**, anch'essa lunga **50** byte.

La successiva funzione **11h**, sempre dell'INT **21h**, serve per ricercare, nella

directory corrente, il *primo* nome di file che corrisponde a quello specificato nell'area fcb. Dal momento che abbiamo deciso di utilizzare il nome *.* (e, quindi, un riferimento a qualsiasi file) il *primo* file presente nella directory interessata verrà trovato. Nel caso in cui la ricerca abbia avuto successo, il registro **AL** viene posto a **0** ed il nome del file trovato inserito dal byte 1 all'**11** dell'area di memoria **dta**. Il nome risulta lungo 11 caratteri (8 di nome + 3 di estensione) senza alcun altro carattere. Se il nome o l'estensione sono più brevi, al posto dei caratteri mancanti saranno presenti degli spazi. Il successivo passo richiama la subroutine **PrintDta**, che scrive su video il contenuto dei byte da 1 a 11 dell'area DTA. Il formato di scrittura è...

```
12345678.123 <CR> <LF>
```

...ossia scrive gli 8 caratteri del nome, il punto (per separare il nome dall'estensione), i 3 caratteri dell'estensione ed infine una sequenza CR - LF per passare sulla riga successiva. A questo punto, tramite la funzione 12h dell'INT 21h, si ricerca il prossimo file corrispondente al nome scelto. Dato che il nome è ancora *.*, sarà ricercato il file successivo presente nella directory.

All'uscita della funzione, AL varrà 0 se il file è stato trovato, ed in questo caso il suo nome verrà messo, come prima, nei byte da 1 a 11 dell'area dta.

Si ripete quindi il ciclo, scrivendo su video l'area dta e cercando un nuovo file.

Se all'uscita della funzione 11h o 12h, INT 21h, AL non vale 0, significa che non sono stati trovati altri file, quindi che la scansione della directory ha avuto termine.

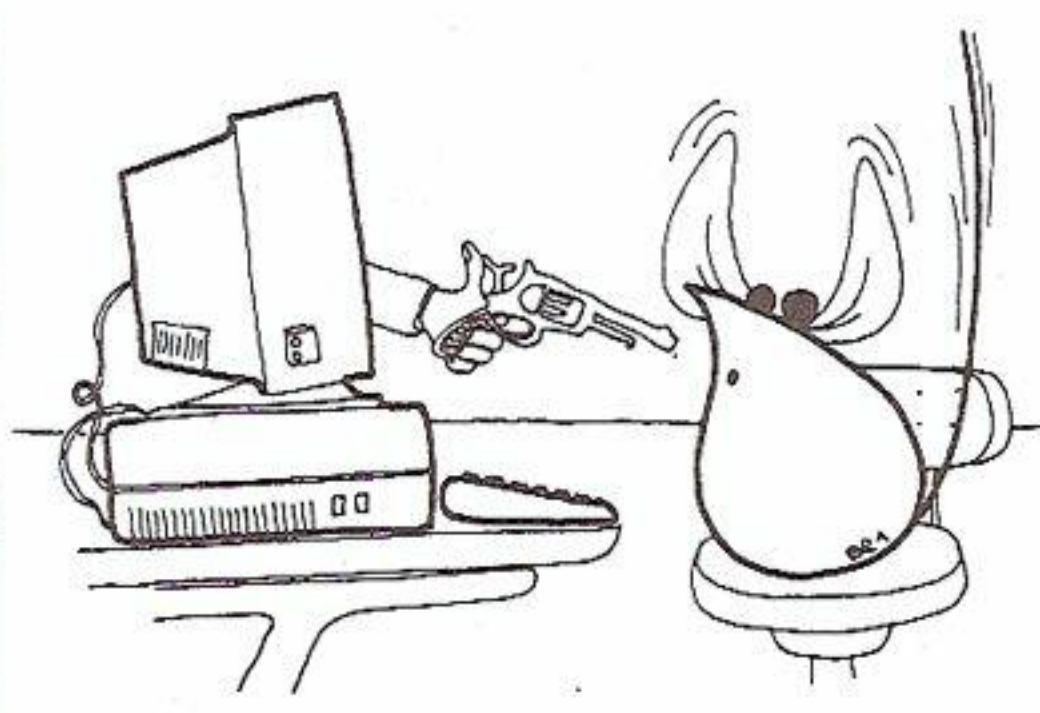
In questo caso, il programma termina e di ritorna al DOS tramite INT 21h, funzione 4Ch.

Digitate il programmino, registratelo con il nome **LIST.ASM** e compilatelo tramite le istruzioni:

```
MASM LIST;
LINK LIST;
EXE2BIN LIST.EXE LIST.COM
DEL LIST.EXE
```

Richiamando il programma da DOS con il comando **List**, potremo renderci conto del suo corretto funzionamento perché apparirà su schermo la lista dei file presenti nella directory correntemente selezionata del disco attivo.

```
;
; LIST.ASM:     Scrive sul video la lista dei file presenti nella directory
;               corrente.
;               NB : Sono escluse dalla lista le eventuali subdirectory
;
cseg          SEGMENT PARA PUBLIC 'CODE'
              org 100h
              ASSUME cs:cseg, ds:cseg, ss:cseg, es:cseg

Start:
              mov   AX,CS              ;Inizializzazioni
              mov   DS,AX              ;DS = CS
              mov   ES,AX              ;ES = CS

              mov   AH,1Ah             ;Setta l'indirizzo dell'area DTA
              mov   DX,offset dta
              int   21h

              mov   AH,29h             ;Prende il nome del file e costruisce il FCB
              mov   SI,offset match
              mov   DI,offset fcb
              mov   AL,0
              int   21h

              mov   DX,offset fcb      ;Cerca il primo file specificato da FCB
              mov   AH,11h
              int   21h
              cmp   AL,0               ;Se non ha trovato niente, finisce
```

```
            jne  FineProg

DirLoop:
            call PrintDta           ;Scrive su video il nome del file trovato
            mov  DX,offset fcb      ;Serca il prossimo file
            mov  AH,12h
            int  21h
            cmp  al,0               ;Se l'ha trovato, ripete il ciclo
            je   DirLoop

FineProg:
            mov  AH,4ch             ;Ritorno al DOS
            mov  AL,0
            int  21h


;
;PrintDta:  Scrive su video l'area DTA. Scrive 8 caratteri, ".", 3 caratteri
;
PrintDta    PROC NEAR
            mov  BX,1               ;Puntatore iniziale
prdtal:
            mov  AL,DS:[dta+BX]     ;Carattere letto dall'area puntata da BX
            push BX                 ;salva puntatore
            int  29h                ;Scrive carattere
            pop  BX                 ;recupera puntatore
            inc  BX                 ;incrementa puntatore
            cmp  BX,9               ;se è arrivato a 9
            jne  prdta2
            push BX
            mov  AL,'.'             ;Scrive un "."
            int  29h
            pop  BX
prdta2:
            cmp  BX,12              ;Continua per 11 caratteri
            jne  prdtal
            mov  AL,10              ;Scrive CR-LF
            int  29h
            mov  AL,13
            int  29h
            ret
printdta    ENDP

;
; Dati usati dal programma
;
            dta      db 50 DUP (0)    ;Area per memorizzazione nome file
            fcb      DB 50 DUP (0)    ;Area per memorizzazione nome ricerca
            match    db '*.* ',0      ;Primo nome da cercare.
;NB: mettere uno spazio
                                      ;alla fine del nome come
                                      ;separatore.

cseg        ENDS
            END  Start
```

Audio, video, mouse e... copiatori

< di Domenico Pavone >

< Una vetrina hardware per utenti Amiga ed MS - DOS >

BMC Cordless Mouse

Riferire qualcosa di originale, quando si tratta di **mouse**, non è impresa facile.

La predilezione per un tocco più o meno rigido sui pulsanti, o la spesso impercettibile differenza di scorrimento sul piano di lavoro, possono infatti essere considerati elementi sì importanti, ma tutto sommato soggettivi. Se non si scomodano periferiche un po' particolari quali trackball, penne ottiche e similari, in fondo (sconvolgente rivelazione in arrivo) un mouse rimane pur sempre... un mouse.

Un primo elemento di novità, seppur relativo, lo si è evidenziato qualche tempo fa (CC n. **83**) descrivendo le caratteristiche di un mouse con rilevazione ottica del movimento.

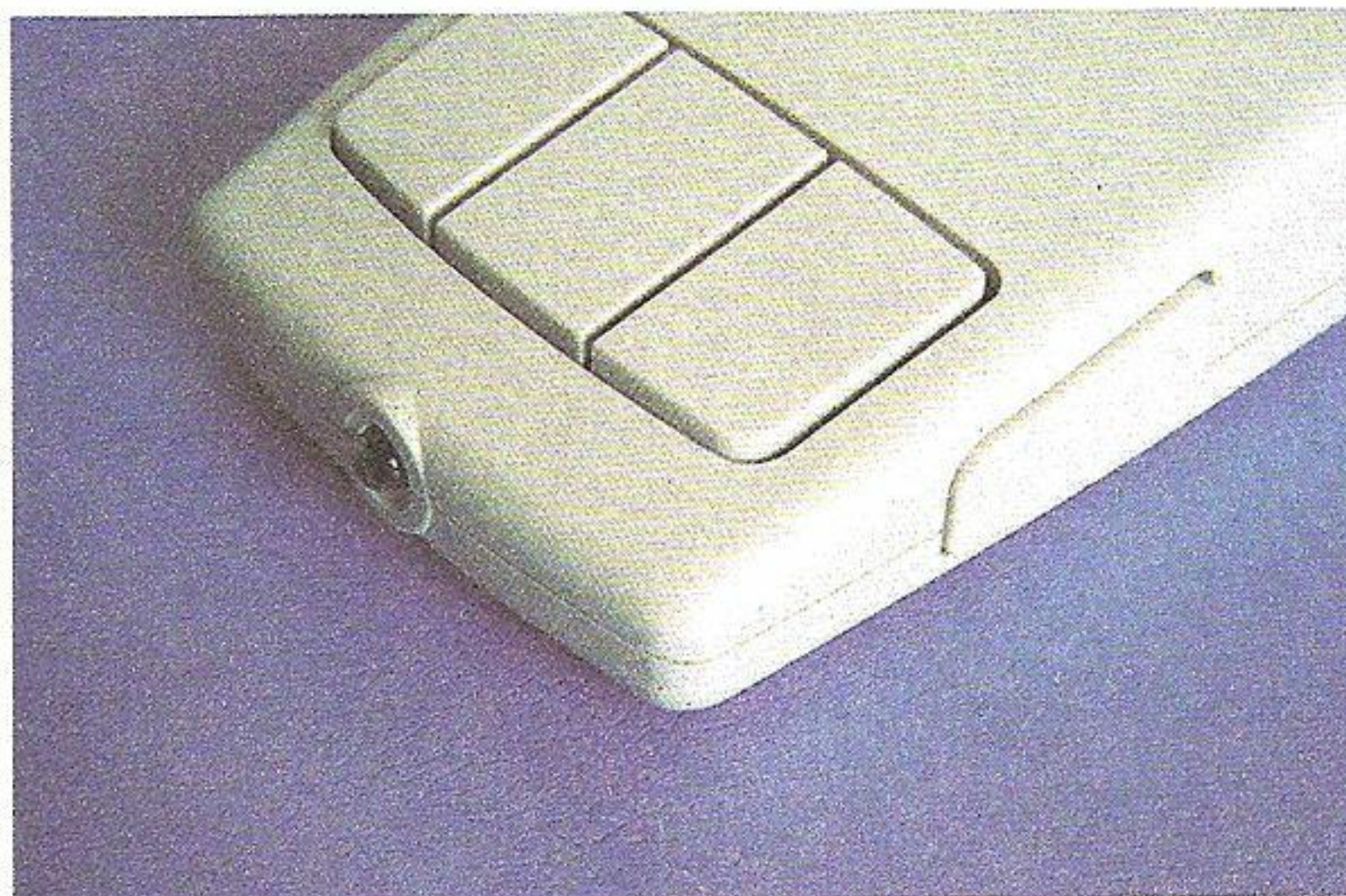
Stavolta ci troviamo di fronte ad una periferica decisamente particolare, che tuttavia non stravolge la consuetudine d'uso dell'accessorio principe di Amiga.

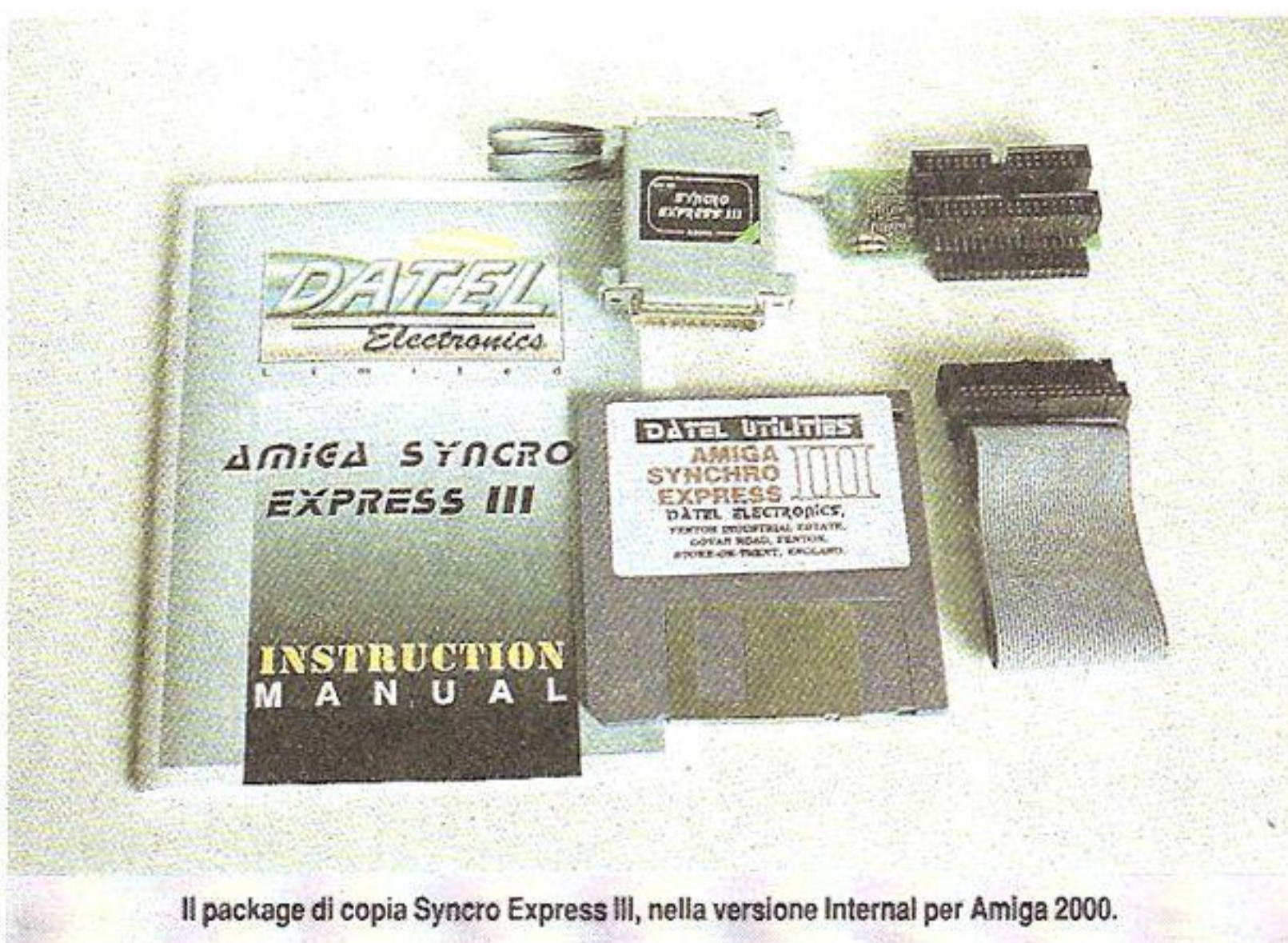
Il **Bmc mouse**, infatti, presenta una caratteristica fondamentale: non necessita di connessione fisica al computer.

Grazie ad una trasmissione di **segnali all'infrarosso** (sullo stile dei telecomandi tv, per intenderci), si può quindi dire finalmente addio ai ben noti aggrovigliamenti dell'odiato cavo sul piano di lavoro, o ai fastidiosi stop della freccetta sul video, legati ad altrettanto frequenti bloccaggi dello stesso contro gli angoli del computer.

Per non dire dei disastrosi crolli di pile di dischetti, di solito strategicamente (sigh) disposti a portata di mano... e di mouse.

Pur se già allettante, non è comunque solo questa caratteristica che rende appetibile Bmc mouse. Vediamolo più da vicino.

Il **Cordeless** (che sta per "senza cavo") è costituito da due componenti hardware: l'elemento ricevente, ed il mouse vero e proprio.

Mouse Bmc. In evidenza il tasto laterale di restart e l'unità di trasmissione del segnale.

Il mouse Bmc Cordless e la sua unità ricevente, da collegare a qualunque modello Amiga

Il package di copia Syncro Express III, nella versione Internal per Amiga 2000.

L'utilizzo su Amiga non richiede alcun intervento software, per cui è sufficiente collegare alla consueta porta di input il connettore di cui è fornita l'unità ricevente, ed il mouse sarà pronto per l'uso. L'unità principale sfrutta come fonte di energia due comuni pile a stilo (preferibilmente alcaline) da 1.5 Volt, fornite a corredo e di comunissima reperibilità, che trovano posto in un alloggiamento sulla superficie inferiore del mouse, ac-

Si ringrazia la Ditta...
Flopperia
Viale Monte Nero, 15
20135 - Milano
tel. (02) 55180484
...per aver messo a nostra disposizione: BMC mouse, Sampler +, Syncro Express III

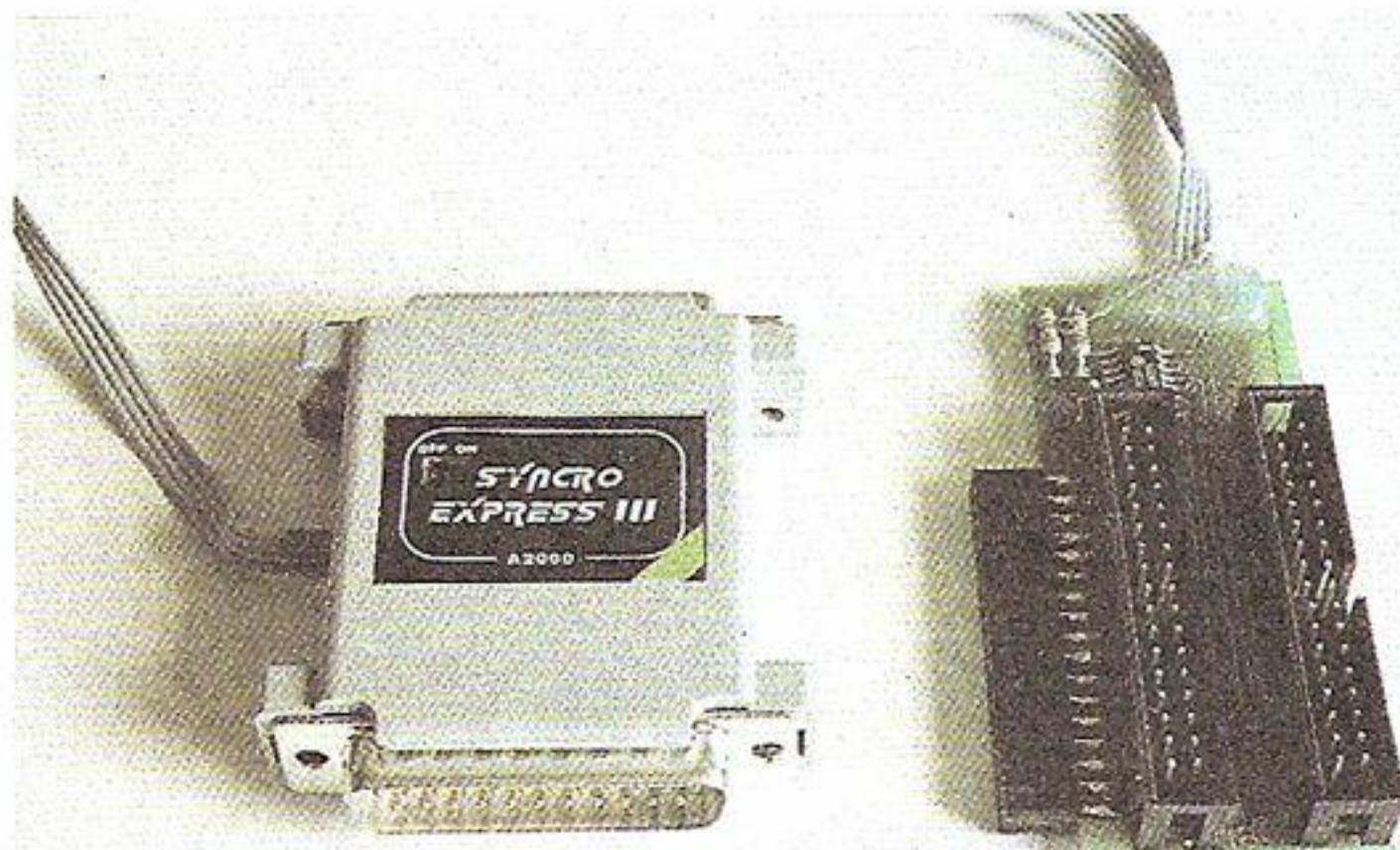
canto alla consueta sfera che trasmette il movimento.

Il mouse è dotato di **tre tasti** sulla superficie di appoggio della mano, mentre un altro si trova sulla faccia laterale sinistra.

Dei comuni pulsanti di "click", in realtà, sono sfruttabili solo i due disposti alle estremità, mentre quello centrale è probabilmente una eredità del modello riservato ai Pc Ibm compatibili, per i quali (come del resto per Atari e Macintosh) è disponibile una versione analoga di Bmc Cordeless.

Particolarmente interessante, invece, la **doppia funzione** svolta dal pulsante laterale.

Questo funge infatti da *interruttore di avviamento*, e da *regolatore della velocità* di scorrimento sullo schermo del puntatore del mouse. In rapporto alla prima funzione va quindi pressato al momento del primo utilizzo, all'accensione del computer, o comunque per riattivare il mouse, che è fornito di uno *stand by* automatico.

Quando lo si adopera, infatti, il Bmc mouse utilizza una certa energia per trasmettere il raggio infrarosso all'unità ricevente, che segnala ogni spostamento accendendo un led rosso di cui dispone frontalmente. Anche quando non lo si utilizza, d'altra parte, si ha un consumo (seppur inferiore) di energia, con conseguente deterioramento delle pile.

Particolare dei connettori di Syncro Express III riservati ad Amiga 2000 con due drive interni.

Per prolungarne la durata, dopo 5 secondi di inattività il mouse entra in uno stato di attesa per così dire parziale, automaticamente sospeso non appena viene rilevato un nuovo movimento.

Se, invece, l'inattività si protrae per 20 minuti, il mouse si "spegne" del tutto. In questo caso, per riprenderne il controllo, si renderà necessario clickare sul pulsante laterale.

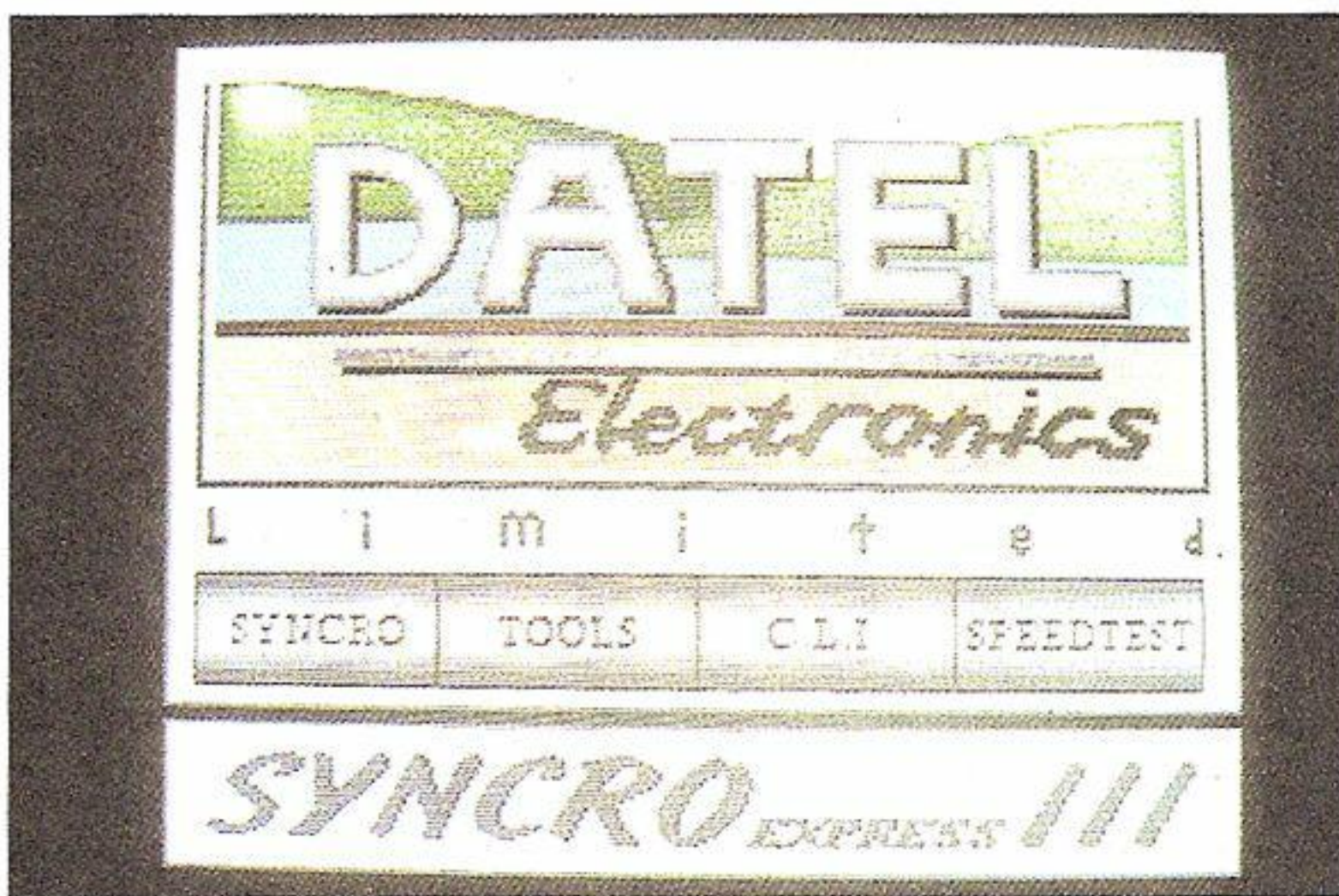
Premuto mentre si eseguono normali manovre di spostamento, lo stesso pulsante controlla invece la velocità del pointer sullo schermo.

Se il mouse viene mosso lentamente, la freccetta si sposterà a piccoli passi, mentre aumentando la velocità del movimento si otterrà uno spostamento a grandi passi del pointer. Decisamente comodo, non c'è che dire.

Sotto un aspetto più squisitamente tecnico, il Bmc mouse ha un raggio di azione che si estende fino ad **1 metro e mezzo** dal ricevitore, con una escursione di 45 gradi a destra o a sinistra.

Come dire che non lo si potrà usare standosene sdraiati sul letto della camera accanto (eventualità peraltro improbabile...), ma una escursione di 90 gradi a un metro e mezzo di distanza è più che sufficiente per qualunque tavolo di lavoro.

Si potrebbe aggiungere che la velocità massima di movimento è di **600 millimetri al secondo** con una risoluzione di **200 dpi**, ma si fa prima a dire che nell'uso non si rilevano sostanziali differenze di prestazione in rapporto ai comuni mouse, con in più la comodità dell'assenza di cavo, e le facilitazioni legate all'uso del pulsante laterale.

Una nuova specie di "ratto" è dunque pronta a invadere i dintorni del nostro computer. Scattante e agile come la precedente, per nulla da estirpare, ma... senza coda!

Lo schermo-menu di Syncro Express III per Amiga.

Syncro Express III

L'esigenza di conservare copie di sicurezza del software originale, e quindi la necessità di riprodurlo, si scontra notoriamente con l'opposta tendenza delle software house a proteggersi dalle duplicazioni clandestine, ricorrendo (soprattutto nel settore games) a **protezioni** spesso insormontabili.

Senza soffermarci in considerazioni di tipo "etico" (e troppo spesso moralistico), chi ne fa le spese è, nella maggior parte dei casi, proprio l'utente medio, che in caso di guasti si troverebbe costretto a salti mortali per riottenere una copia ufficiale del software.

Per la serie *backup facile e totale*, ecco a suo beneficio un package che rappresenta senza dubbio l'evoluzione massima della specie, disponibile praticamente per ogni tipo di computer: Amiga e Pc da un punto di vista strettamente hardware, ma (soprattutto) nel primo caso con la possibilità di trattare floppy anche di diverso formato.

Il copiatore Amiga Syncro Express III al lavoro.

Per Amiga...

Già in passato ci siamo occupati di **copiatori** dotati di ausilio hardware (vedi C.C. n. **78**), ma la **Datel** ha ora provveduto a coprire l'intera fascia di modelli Amiga, quale che sia la configurazione attiva.

Inutile precisare che il backup (o copia che dir si voglia) è riservato al trasferimento di dati da un floppy disk all'altro, mantenendo integre le caratteristiche del supporto.

In altre parole, riproducendo anche formati **non** conformi ad Amiga dos: vuoi per protezione da eventuale copia (...), vuoi perché il floppy è stato formattato da un altro sistema (Atari, Ms-Dos, eccetera).

L'elemento hardware di Syncro Express è costituito fondamentalmente da una specie di riduttore fornito alle estremità di due connetori a **23 pin**, uno maschio e l'altro femmina, da inserire nella porta Amiga riservata ai drive esterni. Nella versione "normale" del package, questo significa che la copia avverrà sempre e solo tra un drive interno ed uno esterno, anche nel caso si possegga un Amiga 2000 con due drive installati internamente.

Per chi volesse, esiste però anche una versione denominata **Internal**, adatta a quest'ultima configurazione, mostrata in dettaglio nelle foto di queste pagine.

All'accessorio prima descritto, in questo caso si affianca una basetta dotata di tre connettori ad essa solidali.

In pratica, occorrerà aprire il cabinet del computer, scollegare il cavo a nastro inserito nel drive **df0:** e sostituirvi il connettore laterale della basetta.

Un altro cavo fornito in dotazione al package farà da *ponte* tra **df0:** e **df1:** mentre il riduttore andrà sempre collocato all'uscita dei drive esterni di Amiga 2000, rimuovendo una delle piastre metalliche poste sul retro per consentire il passaggio del cavo di connessione tra riduttore e piastrina.

Più facile a farsi che a dirsi, l'operazione non richiede che un paio di minuti. In questa configurazione, si potrà in pratica

La versione per Pc Ibm compatibili di Syncro Express III.

disporre comunque di un drive esterno (df2:), mentre le operazioni di copia con Syncro Express III potranno essere svolte utilizzando i due drive interni del 2000.

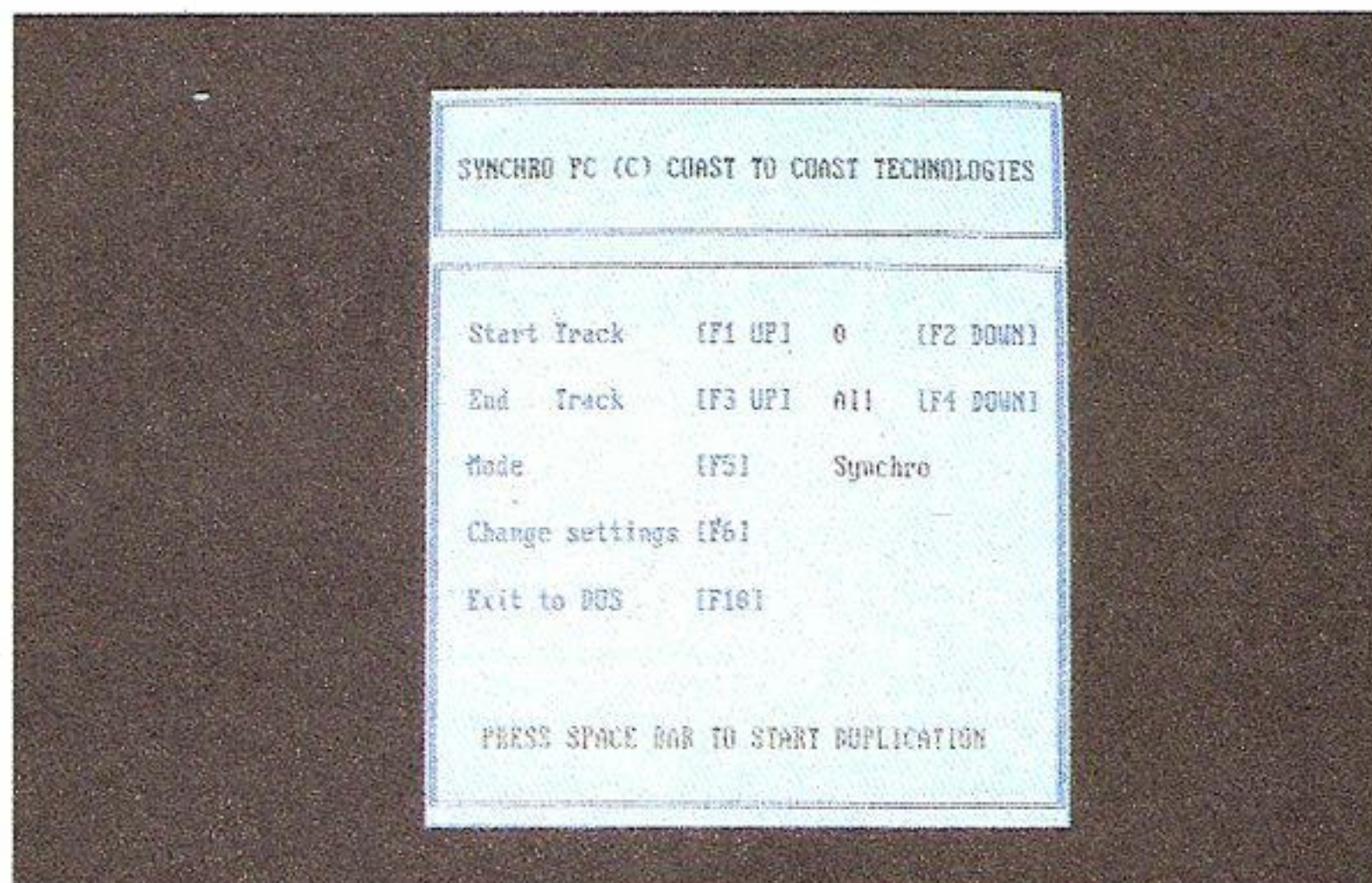
Il backup andrà effettuato, come ovvio, sotto stretto controllo software, che peraltro potrà essere sfruttato tanto da solo che in congiunzione con l'hardware appena descritto.

Il connettore principale, infatti, dispone di un **microinterruttore** on/off, che può disabilitare Syncro Express anche in fase di copia, nel caso si decida per una tecnica "mista" o esclusivamente software. Vediamo di chiarire l'arcano.

Una volta lanciato in autoboot il floppy in dotazione al package, si accede a una prima schermata grafica che funge da menu principale.

Attraverso le sue opzioni, è possibile attivare uno **Speedtest** per valutare la velocità di accesso ai dati di un floppy, nonché una classica utility (a nome **Tools**) sullo stile di *DirMaster*, *Climate* e similari, per gestire in modo intuitivo le più comuni operazioni su files e directory.

Scegliendo, invece, la voce **Syncro** dallo stesso menu, si accede al tool di backup vero e proprio.

Da questo ambiente è possibile modificare la traccia di inizio e fine della copia (fino a **90 tracce!**), e impostarne la modalità. In particolare, agendo sul pulsante destro del mouse, si potrà decidere se

Lo schermo di lavoro di Syncro Express III per Pc, molto più spartano che nella versione Amiga

effettuare una copia **Normal**, **Index** oppure **Syncro**.

Sorvolando sull'ovvio significato dell'opzione Normal, con Index si rende possibile la copia di *qualunque* formato, ivi compresi quelli non Amiga.

La scelta Syncro, invece, attiva un controllo sulla velocità dei due drive interessati alla copia, tentando una riproduzione del floppy di origine dettagliata sino all'inverosimile. Quest'ultima modalità, c'e' da dire, risulta (ovviamente) molto più lenta delle precedenti, che riescono ad ultimare una copia in soli **50 secondi**(!).

Quanto detto, è da considerarsi valido se l'hardware di Syncro Express è attivo. Se, invece, si pone lo switch del riduttore in posizione **off**, la modalità Syncro del software diventerà un più comune **Nibble Copy**, grazie alla capacità del programma di valutare la presenza o meno dell'hardware.

In particolari casi in cui il tool rileva l'impossibilità di riprodurre fedelmente alcune tracce del disco origine, provvede esso stesso a fare un ultimo tentativo, chiedendo espressamente che si disattivi l'hardware, per poi procedere in modo Nibble.

Caso in realtà molto, molto raro.

Il digitalizzatore audio Sampler+ con cavetto stereo in dotazione e manuale in italiano.

...e per PC

Ciò che sui computer della classe Amiga è... pane quotidiano, ovvero la protezione del software ottenuta ricorrendo a metodi fisici sui floppy, in ambito PC e compatibili è molto meno frequente.

Ciò nonostante, un package di backup come **Syncro PC** (è questo il nome della versione PC della Datel) può ugualmente risultare appetibile, grazie soprattutto a prestazioni normalmente non accessibili al DOS.

Anche in questo caso ci si trova di fronte a un corredo hard/soft, costituito essenzialmente da una scheda da inserire in uno slot del proprio PC IBM compa-

Vista frontale di Sampler+. Si notino gli ingressi per i 2 canali stereo e la manopola per regolare il livello di input.

tibile, che opera sui due drive **a:** e **b:**. La compatibilità con diversi tipi di macchina è garantita da una serie di microswitch configurabili in modo da adattarsi a IBM, Epson, Tandy, Amstrad, e "cloni" generici.

Una volta installata, in uno slot più vicino possibile ai drive, ed effettuati pochi e semplici collegamenti molto chiaramente illustrati nel manualetto a corredo, il software di Syncro PC consente di operare le scelte desiderate attraverso l'uso dei **tasti funzione.** La precisazione sul tipo di compito ad essi assegnati è mostrata a video in uno spartano (ma chiaro) riquadro di lavoro.

Premesso che l'operazione di copia è decisamente **più veloce** che adoperando il normale **diskcopy**, la caratteristica più interessante del package riguarda il formato dei floppy.

La copia può infatti esere eseguita (come ovvio) con dischi entrambi da 5.25" oppure entrambi da 3.5", anche in alta densità, ma è altresì possibile effettuare il backup di un disco da 5.25 in uno da 3.5, prestazione, questa, davvero inconsueta. E' anche possibile selezionare un ristretto numero di tracce da riprodurre (tasti F1 - F4) o anche limitare la copia ad una sola delle facciate, ma di norma l'unica opzione di cui si farà uso riguarderà la modalità di copia: **Synchro** (di default), **Synchro HD** per dischi ad alta densità, oppure **Super 1** e **Super 2**.

Queste ultime, in pratica, andranno utilizzate solo in caso di particolari formati non ammessi dal DOS, come per esempio floppy formattati a **42 tracce**, oppure contenenti settori nascosti a scopo di protezione: se l'operazione non andrà felicemente in porto adoperando il primo, si passerà al secondo.

Se anche in questo caso l'esito non fosse soddisfacente, vorrebbe dire che... si è dimenticato di installare Syncro Pc.

Sampler+

Nell'ipotetica rassegna di **digitalizzatori audio** per Amiga, diventata via via una consuetudine per renderne più semplice l'eventuale scelta di acquisto, Sampler+ della **P.G. Elettronica** ha tutte le carte in regola per occupare una precisa collocazione.

Di tutto rispetto, naturalmente.

Nei precedenti appuntamenti con questa rubrica abbiamo, infatti ,avuto modo di esaminare modelli di ogni categoria, dai sampler monofonici di bassissimo prezzo, ad altri più cari ma legati a prestazioni per così dire "miste", come l'associazione di campionamento stereo e interfacciamento strumentale via Midi.

In una ideale via di mezzo, valida alternativa per chi necessita di alta qualità, può senz'altro porsi Sampler+, un digiti-

zer stereo finalizzato al **solo campionamento audio**, in cui l'assenza di input / output Midi è ripagata da un conseguente (notevole) abbassamento del costo.

E, si badi, quando si parla di qualità ci si riferisce ad un *aggeggio* in grado di campionare frequenze fino a **58 KHz** in monofonia e **38 KHz in stereo**!

Il Sampler+ viene fornito senza un software specifico per pilotarne il funzionamento, e questo per una ragione molto semplice: **è compatibile al 100% con Audiomaster III**, del quale ci siamo occupati a fondo sul non lontano numero **83** della rivista.

Una scelta tra l'altro ben oculata: meglio affidarsi ad un prodotto già esistente, di caratteristiche professionali e divenuto in men che non si dica uno standard del settore, piuttosto che cimentarsi nell'ardua impresa di riprodurne l'efficacia.

Esteriormente Sampler+ si presenta con uno chassis nero di dimensioni contenute, fornito sul pannello anteriore e posteriore di due comuni connettori Rca per l'input/output del segnale sonoro.

In particolare, gli ingressi posteriori vanno collegati (con un cavetto stereo fornito in dotazione) alle **porte audio di Amiga**, mentre sul frontalino andranno inseriti i connettori provenienti dalla fonte sonora.

La facciata anteriore comprende anche un mini jack per un eventuale ingresso da **microfono**, nonché una manopola di regolazione del livello di input, che così non sarà necessario dosare direttamente all'origine, spesso costituita da un apparecchio disposto lontano dal computer.

Altro particolare degno di nota, Sampler+ dispone di una **uscita amplificata** per cuffie stereo (con impedenza non inferiore a 32 ohm), per cui è possibile ascoltare in diretta l'andamento del suono. L'interfacciamento fisico con Amiga, infine, è realizzato attraverso le porte seriale e parallela, nelle quali va inserito l'opportuno connettore proveniente dal digitalizzatore.

Nell'uso con Audiomaster III, occorrerà prima di ogni altra cosa selezionare l'opzione **Hi-Speed** del menu *Options/Sampler Configuration* (peraltro di default), evitando accuratamente le altre scelte, che modificano lo status della porta parallela.

Il campionamento vero e proprio andrà poi eseguito come di norma, a cominciare dalla regolazione del volume di input aiutandosi con la visualizzazione ottenuta tramite il gadget *Monitor* del programma, nel caso di Sampler+ con la comodità supplementare della sua manopola Level.

Cui farà seguito, come ovvio, una bella clickata su Sample, una volta deciso il brano da registrare.

Come precisato dalle istruzioni allegate all'hardware, Sampler+ provvede

Il pannello posteriore di Sampler+, che comprende un connettore per ascolto in cuffia.

automaticamente ad eliminare il filtro passa-basso implementato da Amiga, con conseguente esaltazione dei toni acuti.

Nel caso lo si desiderasse, questo può comunque essere attivato via software da Audiomaster (menu Options/Audio Filter).

Per campionamenti monofonici, basterà associare, alla relativa selezione Mono su Audiomaster, l'uso del solo canale destro (Right) su Sampler+.

Cosa peraltro consueta nelle fasi di digitalizzazione, Sampler+ si dimostra particolarmente sensibile ai disturbi indotti da un cattivo schermaggio dei cavi adoperati, o da eccessiva vicinanza degli stessi ad apparechiature elettriche come alimentatori (quello di Amiga non fa eccezione), tv, monitor, eccetera.

Una certa accuratezza eviterà comunque ogni problema, e la pulizia del suono rallegrerà sicuramente qualche floppy Amiga dall'aria prima triste...

Rocgen RG300C

Dopo numerosi appuntamenti con la elaborazione di video amatoriali, il lettore attento conoscerà ormai certamente il significato del termine **Genlock**.

Rocgen, un economico ma valido genlock della Roctec.

Se ancora perdurasse qualche dubbio, ricorderemo comunque che si tratta di un apparecchio che consente di miscelare il segnale video di Amiga a quello proveniente da un'altra fonte come videotape, telecamere, o lo stesso apparecchio tv. In tal modo, riversando il risultato della miscelazione su un altro tape, o anche semplicemente redirigendolo verso un monitor per una semplice visualizzazione, si possono ottenere risultati simili a quelli cui ci hanno abituati gli innumerevoli effetti speciali presenti negli spot pubblicitari, o più banali titolazioni (e abbellimenti) delle proprie sceneggiature (alla faccia della modestia).

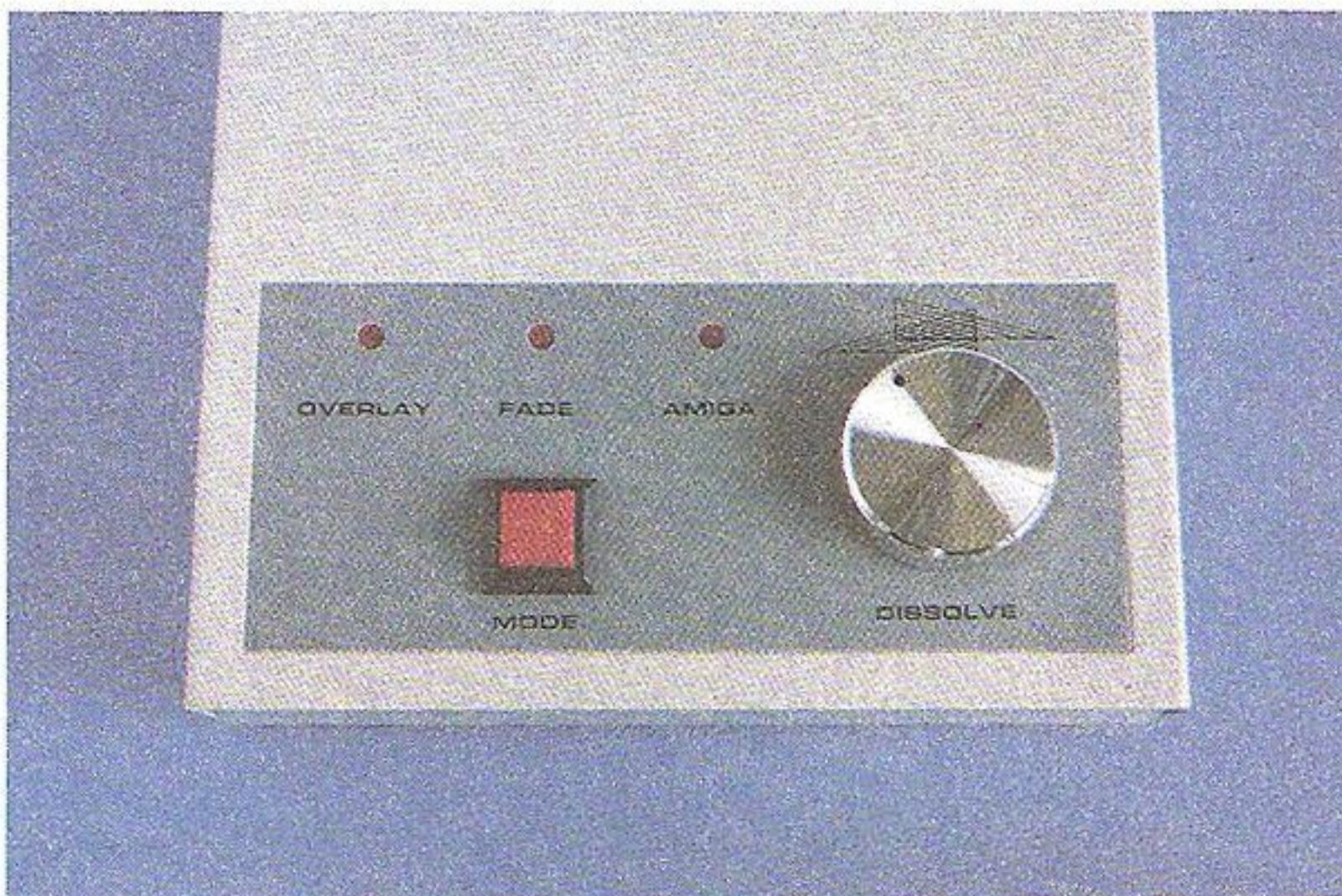
E fin qui tutto semplice, se non fosse per un dettaglio non insignificante: l'alto costo che normalmente raggiungono simili attrezzature.

Come altre volte ribadito, e a patto di sacrificare un po' la qualità complessiva dell'immagine, è possibile accostarsi ad hardware più economico se ci si accontenta della gestione in videocomposito del segnale.

In questa fascia di utenza le presenze commerciali sono ormai numerose, per cui RocGen va ad aggiungersi ad un settore già abbastanza affollato.

Con una chance in più, però: il **bassissimo prezzo**, nonostante un livello accettabile di prestazioni, certamente non inferiore ai colleghi della stessa categoria.

Questo genlock si presenta con un'estetica *slim line* che lo fa somigliare ai drive per floppy disk, dei quali emula anche la dimensione.

E' fornito di un pannello di controllo sulla faccia superiore che comprende un pulsante sovrastato da tre spie luminose, ed una manopola. Sul posteriore, sono invece presenti due prese Rca per l'input e l'output del segnale video composito, ed un cavo dotato di connettore standard a **23 pin** che va inserito alla porta video di Amiga. Come ovvio a computer spen-

Il pannello di controllo di Rocgen, posto sulla faccia superiore dell'accessorio.

to, e dopo aver rimosso il cavo di connessione tra computer e monitor.

Eseguita questa banale operazione, va poi effettuato il collegamento con la fonte video. Di solito, va detto, le apparecchiature moderne sono dotate di **presa Scart**, per cui si renderà necessario procurarsi (o fare da sé) un cavo che presenti a un capo un connettore Scart, e all'altro le corrispondenti uscite in videocomposito (di solito sono comprese anche le uscite audio), di facile reperibilità nei negozi specializzati.

Il segnale video (di solito nei cavetti è contrassegnato) andrà connesso alla presa di RocGen denominata **Video In**.

A questo punto tutto è pronto per funzionare, ma è opportuno anche decidere quale sarà la destinazione del segnale: un altro videotape (e quindi, con tutta probabilità, si renderà necessario un altro cavo che dal **Video Out** vada alla Scart dell'apparecchio ricevente), oppure lo stesso monitor di Amiga.

In quest'ultimo caso sarà sufficiente procurarsi un cavetto fornito di due spinotti Rca alle estremità, e collegare l'uscita video del genlock direttamente all'**ingresso CVBS** del monitor. Per visualizzare il segnale basterà agire poi sul pulsantino anteriore dei monitor di solito adottati da Amiga, in modo da visualizzare il segnale videocomposito.

La qualità dell'immagine, si constaterà, non è paragonabile a quella ottenuta dal consueto RGB, ma con opportuni accorgimenti, ovvero evitando eccessi di colori o l'accoppiamento di alcuni, si possono comunque raggiungere risultati accettabili.

I connettori di input/output del genlock Rocgen.

Ultimati i preparativi, si può pasare all'azione: basta attivare Amiga con un disco di start inserito nel drive, nonché la fonte video. Sullo schermo del monitor, sia esso collegato direttamente al genlock oppure al tape ricevente, si potrà ora far giungere il segnale che si preferisce, agendo sul pulsante del pannello di controllo di RocGen.

Ad ogni pressione si selezionerà uno dei tre modi possibili, evidenziati dall'accensione della corrispondente spia luminosa presente sull'apparecchio: **Overlay**, **Fade**, **Amiga**.

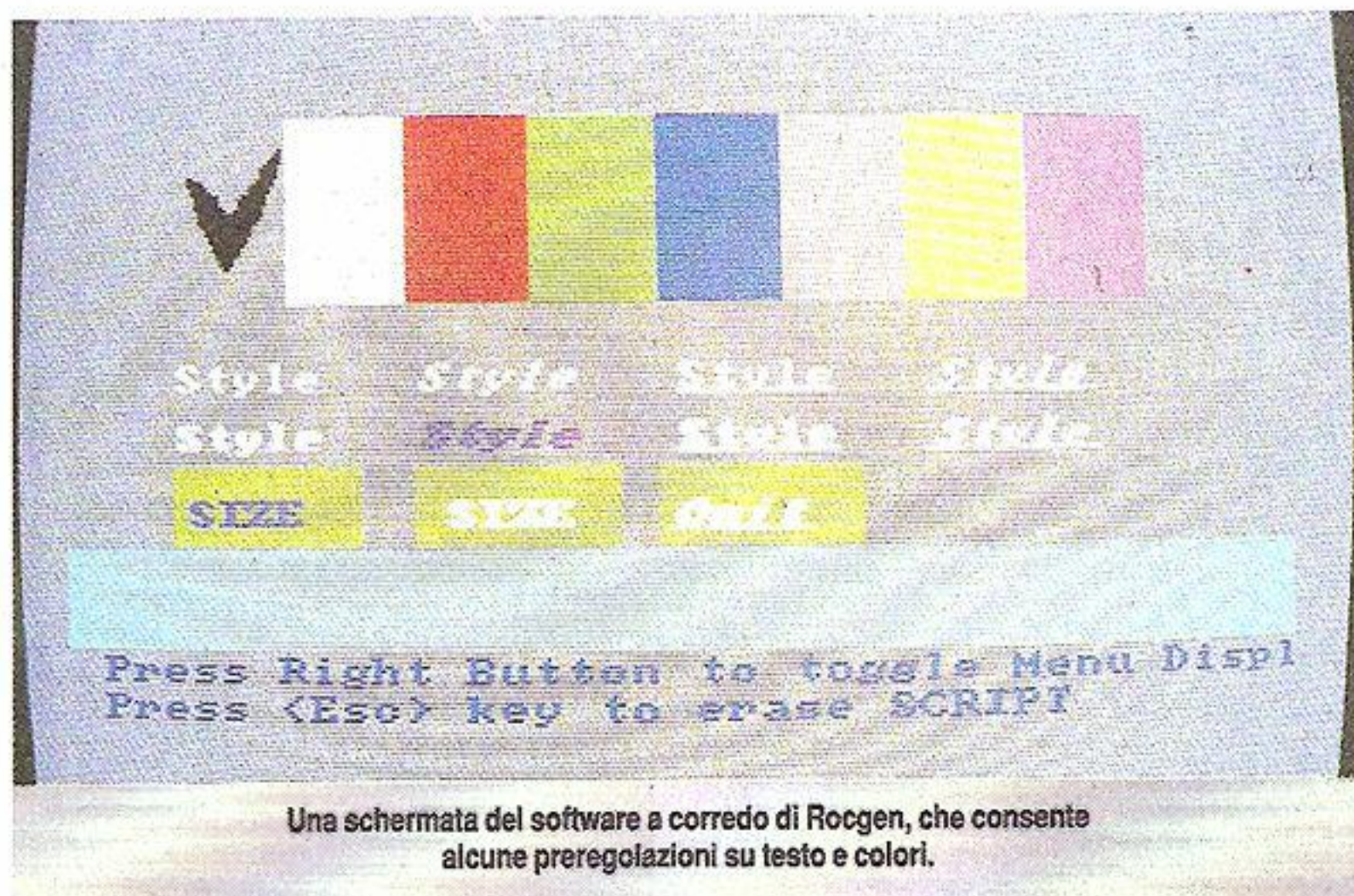
L'ultimo si limita a mostrare il solito schermo del computer, mentre gli altri due permettono, con una intensità regolabile dalla manopola di RocGen, la **miscelazione del segnale**.

Ruotando gradualmente la manopola in un senso o nell'altro, si potrà in pratica attuare una **dissolvenza dello schermo Amiga** fino a farlo scomparire, mentre al suo posto farà bella mostra di sé l'immagine proveniente dalla fonte video.

In alternativa si potrà limitare la dissolvenza ai soli elementi di primo piano dello schermo Amiga, mentre lo sfondo resterà occupato dal segnale esterno.

Senza dimenticare, naturalmente, che per un livello professionale è indispensabile una attrezzatura professionale.

Ma senza dimenticare, anche, che RocGen costa solo **275.000** lire...

Una schermata del software a corredo di Rocgen, che consente alcune preregolazioni su testo e colori.

Si ringrazia la Ditta...
CIRCE Electronics
Via F. Testi, 219 Milano
Tel. 02/ 6427410 (zona Niguarda)
V.le Monza, 6 Milano
Tel. 02/ 26.11.2024 (zona Loreto)
...per aver messo a nostra disposizione Rocgen RG300C

Metti una foto nel tuo computer

< di Carlo d'Ippolito >

< Manipolazione delle immagini riprese >

Da qualche anno la Canon, insieme ad altre aziende del settore, ha iniziato la produzione di una speciale macchina fotografica che non usa comune pellicola, ma speciali floppy disk.

E' destinata ad affiancare la macchina fotografica tradizionale nè più nè meno di come il videoregistratore *affianca* oggi, ma sarebbe meglio dire *sostituisce*, la cinepresa di antica memoria.

Il concetto è molto semplice: se l'obiettivo invia la luce su un sensore a colori CCD da 1/2 pollice (capace di **786 pixel** in orizzontale), verrà generato un segnale in grado di essere elaborato e riversato su un dischetto magnetico abbastanza piccolo da esser contenuto nell'altrettanto minuscola apparecchiatura.

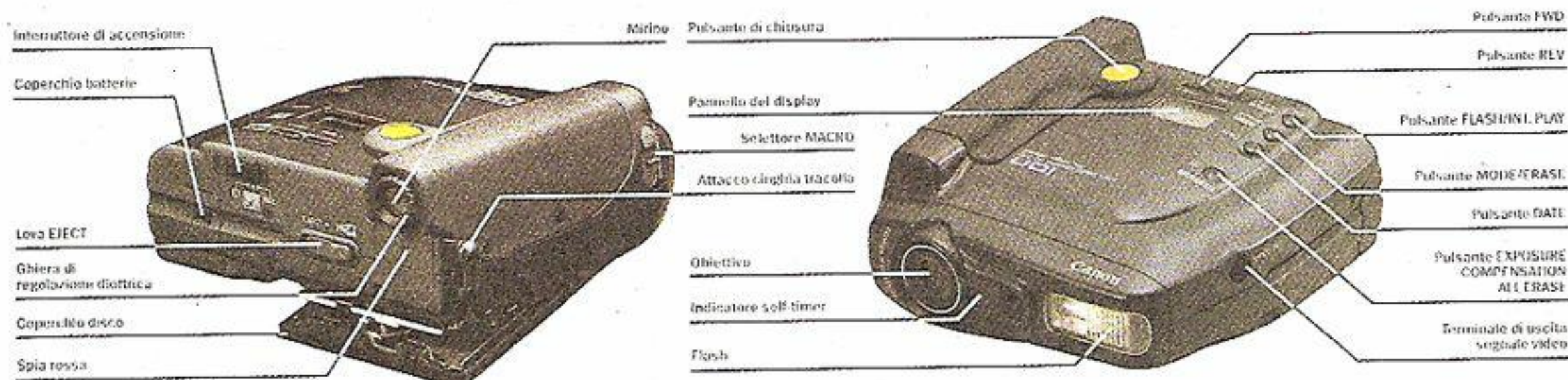
Il problema principale è, appunto, quello di utilizzare in modo opportuno le riprese effettuate.

Alla dotazione della **ION RC-260** appartiene un cavetto che consente il collegamento ad un qualsiasi apparecchio TV color per rivedere il materiale ripreso.

E' ovvio che la qualità non può esser paragonata a quella ottenibile con una diapositiva, ma certamente è del tutto simile a quella che un tradizionale sistema TV moderno consente (esempio: ottima ricezione di TV commerciali; nastri videoregistrati di alta qualità).

Chi è abituato alla qualità del *media* **TV** (del resto ottima) saprà apprezzare questo straordinario apparecchio che consente di memorizzare fino a **50 immagini**, cancellabili e riscrivibili a volontà, per ciascun dischetto.

Il prezzo di partenza può apparire elevato.

Ma se si pensa al ridottissimo costo di esercizio, dovuto alla possibilità di sovrascrivere immagini non desiderate, alla fine dei conti l'operazione può presentare i suoi lati positivi.

Il Kit ION - PC

L'aspetto più interessante offerto dal prodotto Canon, e che ci riguarda più da vicino, è la possibilità di manipolare le immagini riprese anche per mezzo di un personal computer.

Un particolare Kit hardware / software, lo ION-PC, comprende una still video camera ION, una scheda digitalizzatrice video per PC/IBM e compatibili ed il software per facilitare l'import / export dell'immagine su applicazioni tipiche, quali DTP e database multimediali.

Il sistema si può reggere su un calcolatore **80286** e scheda **EGA**, ma viene suggerito almeno un sistema **80386** con VGA o **Super VGA**.

Il prezzo al pubblico è abbastanza salato (**2.300.000** oltre l'I.V.A.) ma è ovvio che una simile apparecchiatura si rivolge ad un'utenza almeno semi-professionale, in grado di ammortizzare in tempi brevi l'investimento effettuato.

Formati grafici

Il software del kit ION-PC è in grado di memorizzare immagini in uno qualsiasi dei seguenti moduli:

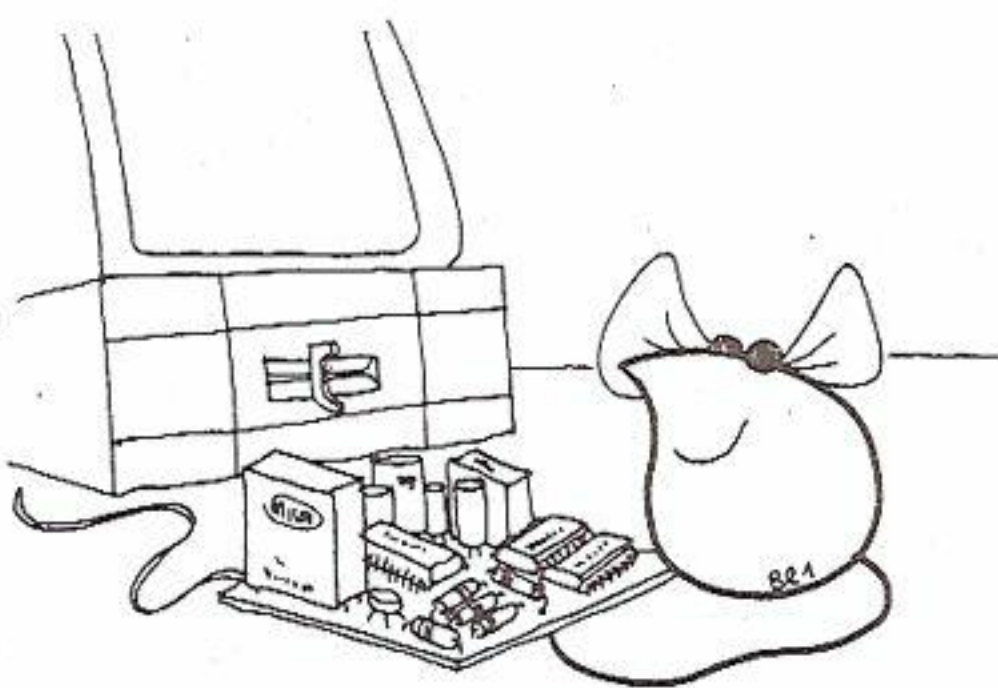
➪ **TI(FF)**. Modulo standard MicroSoft Aldus, utilizzato su sistemi PC ed Apple Macintosh. I file di questo tipo, anche se non provenienti da riprese effettuate con la still video camera, possono esser richiamati dal software per eventuali elaborazioni.

➪ **PCX**. E' il tipico formato **PC** Paintbrush, utilizzabile anche per colorazioni.

➪ **IMG**. E' l'ambiente di tutti i pacchetti che girano sotto GEM (tra cui Ventura) ed è compatibile con il sistema di fotocomposizione Ventura.

➪ **GIF**. Utilizzato nel sistema *Compuserve*, è ampiamente sfruttato per interscambio tra computer e servizi telematici di notizia-ri.

➪ **TGA**. True Vision Targa, altro formato per PC.

➪ **MSP**. Sistema Bit Map a bassa definizione della Microsoft Paint.

I computer

Come già detto, il sistema Canon ION può essere impiegato, se opportunamente interfacciato, con i tre tipi di computer più diffusi attualmente. Parliamo dei sistemi MS - DOS, dei **Macintosh** e dell'**Amiga**, anche se quest'ultimo sembra (per ora) limitato al modello 3000 (grazie al pacchetto multimediale **Amiga Vision** della Commodore). Ci ripromettiamo di provare il kit e di dare notizia dei risultati del test. Il tallone d'Achille di tali apparecchi di ripresa, infatti, è costituito dall'inevitabile scadimento di **qualità** nel passare dallo schermo al programma di utilizzo (tipicamente, un DTP) e da questo alla carta (stampante laser o fotocomposizione). Un analogo decadimento di immagine si verifica anche nel passaggio dell'immagine ripresa su carta fotosensibile a colori. Su questo punto, e per molto tempo ancora, è probabile che il tradizionale sistema di ripresa, basato su procedimenti chimici e pellicole fotoimpressionabili, abbia molte carte ancora da giocare. Però non è detta l'ultima parola: la stessa cosa si diceva della qualità delle riprese magnetiche effettuate con videocamere e pochi avrebbero scommesso sul sorpasso di tali apparecchiature sulle vecchie cineprese superotto o 35 millimetri. E tutti sappiamo come è andata a finire...

Per ulteriori informazioni:
Canon Italia
Divisione Foto video
Via Mecenate, 90
20138 Milano

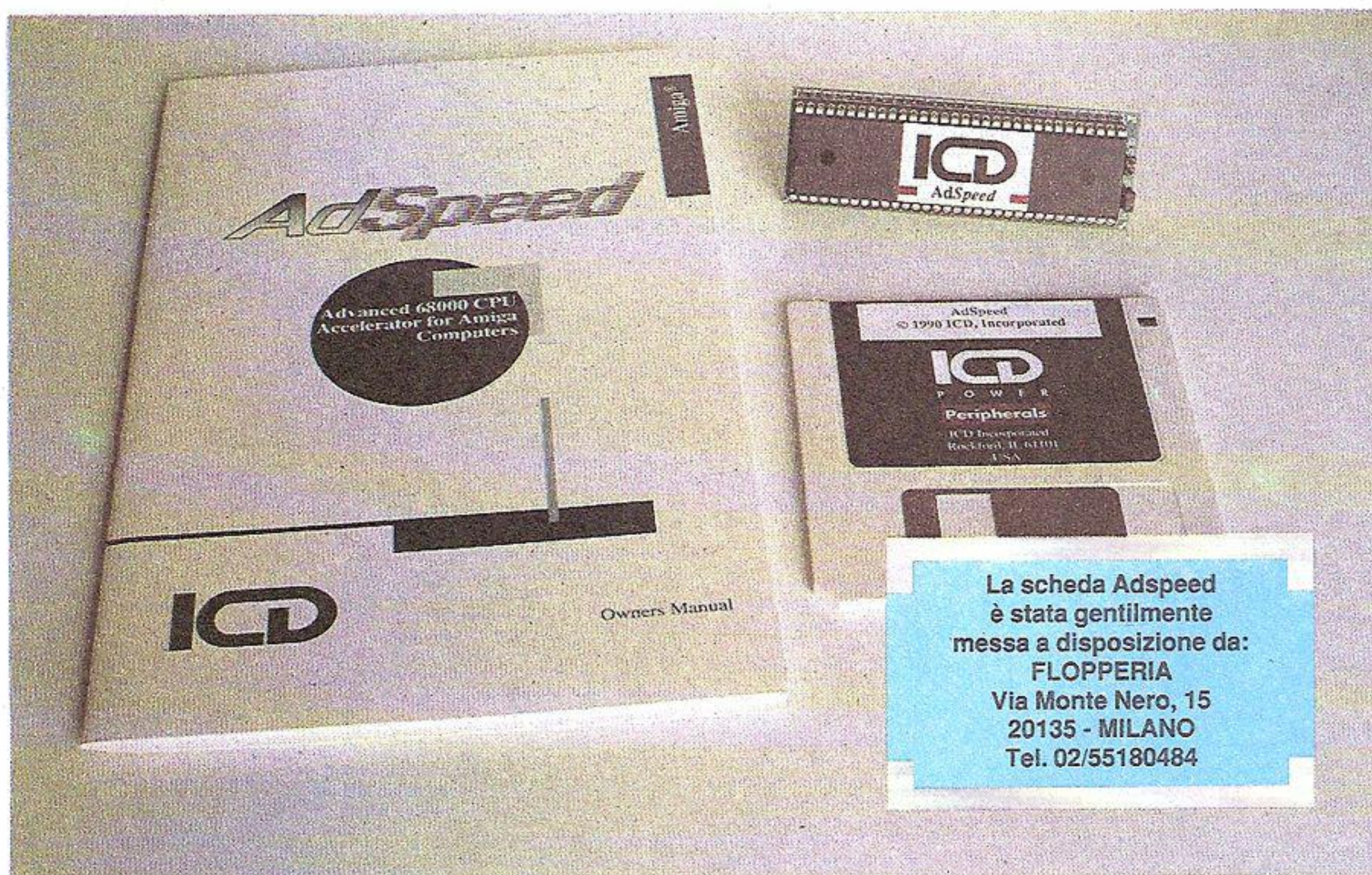
✔ Amiga	❑ Principianti	❑ Esperti	✔ Tutti	✔ HELP

- ❑ Ms - Dos
- ❑ Recensioni
- ✔ Hardware
- ❑ Software
- ❑ Applicazioni
- ❑ Programmazione
  - ❑ Dos
  - ❑ Pascal
  - ❑ C
  - ❑ Basic
  - ❑ Assembly

L'alta velocità è caratteristica di schede costose; qualcuno, però, ha pensato anche agli utenti del modesto Amiga 500 che, in certi casi, possono veder raddoppiata la velocità del proprio computer

Amiga a tutta birra

⇨ < di Domenico Pavone >

< Aumentiamo la velocità operativa del nostro Amiga 500 / 2000 >

La scheda Adspeed
è stata gentilmente
messa a disposizione da:
FLOPPERIA
Via Monte Nero, 15
20135 - MILANO
Tel. 02/55180484

Il corredo hard\soft della mini scheda acceleratrice della Icd per tutti i modelli di Amiga

Sono passati solo pochi anni da quando Amiga ha fatto la sua apparizione sul mercato, stupendo per le sue prestazioni grafiche e sonore, ma anche per quella che sembrava una **velocità di elaborazione** stratosferica: tanto assoluta, quanto in rapporto alla sua frequenza di lavoro, limitata (se così si può dire) a **7.16** Mega Hertz.

Velocità in effetti legata più ad una originale distribuzione del lavoro, che veniva (e viene) svolto solo quando necessario dall'unità centrale (il microprocessore **68000**, per intenderci), mentre i compiti più gravosi quali trasferimenti, grafica, eccetera vengono affidati ai cosiddetti *chip custom*.

Senza addentrarci troppo in dissertazioni tecniche, vanno però fatte due considerazioni. La prima, più banale, che a tutto ci si abitua.

La scheda Adspeed, da inserire al posto del chip 68000 in dotazione ad Amiga

Inoltre, col passare del tempo, il progresso tecnico ha consentito a molti utenti, prima ultra soddisfatti, di toccare con mano le delizie di velocità ancora superiori. Come se non bastasse, ogni Commodore Point che si rispetti tiene in bella mostra un **Amiga 3000**, oggetto di brame inconfessate.

Per chi, a questo punto, non riesce più ad accontentarsi del solito Amiga 500 o 2000, e non intende spendere la cifra certamente non indifferente (e al momento giustificabile solo da applicazioni professionali) per passare al boss 3000, non c'è che una soluzione: *dare una smossa* al proprio, ricorrendo a una scheda acceleratrice.

Anche in questo caso, il problema assume comunque connotati prettamente economici. Disponendo di un A-2000, la soluzione migliore consisterebbe nell'adottare una scheda **68030**, in grado di colmare il gap con Amiga 3000.

Chi, invece, non ha bisogno di velocizzazioni ultra, vuole spendere cifre modeste, e magari dispone di un A-500, può sempre ricorrere a schede che si limitano ad accelerare il clock e alcuni accessi alla ram. Vediamo dunque due prodotti che, in accordo con quanto detto, riassumono le due possibili alternative.

Adspeed vista dal lato dei componenti. Si notino le ridottissime dimensioni dell'insieme.

Adspeed

Prodotta dalla statunitense **Icd**, nota per aver introdotto sul mercato accessori particolari come il **Flicker Free Video** o un mini **hard disk interno** per Amiga 500, AdSpeed è una scheda dalle dimensioni ridottissime, adattabile a tutti i modelli (3000 escluso, ovvio).

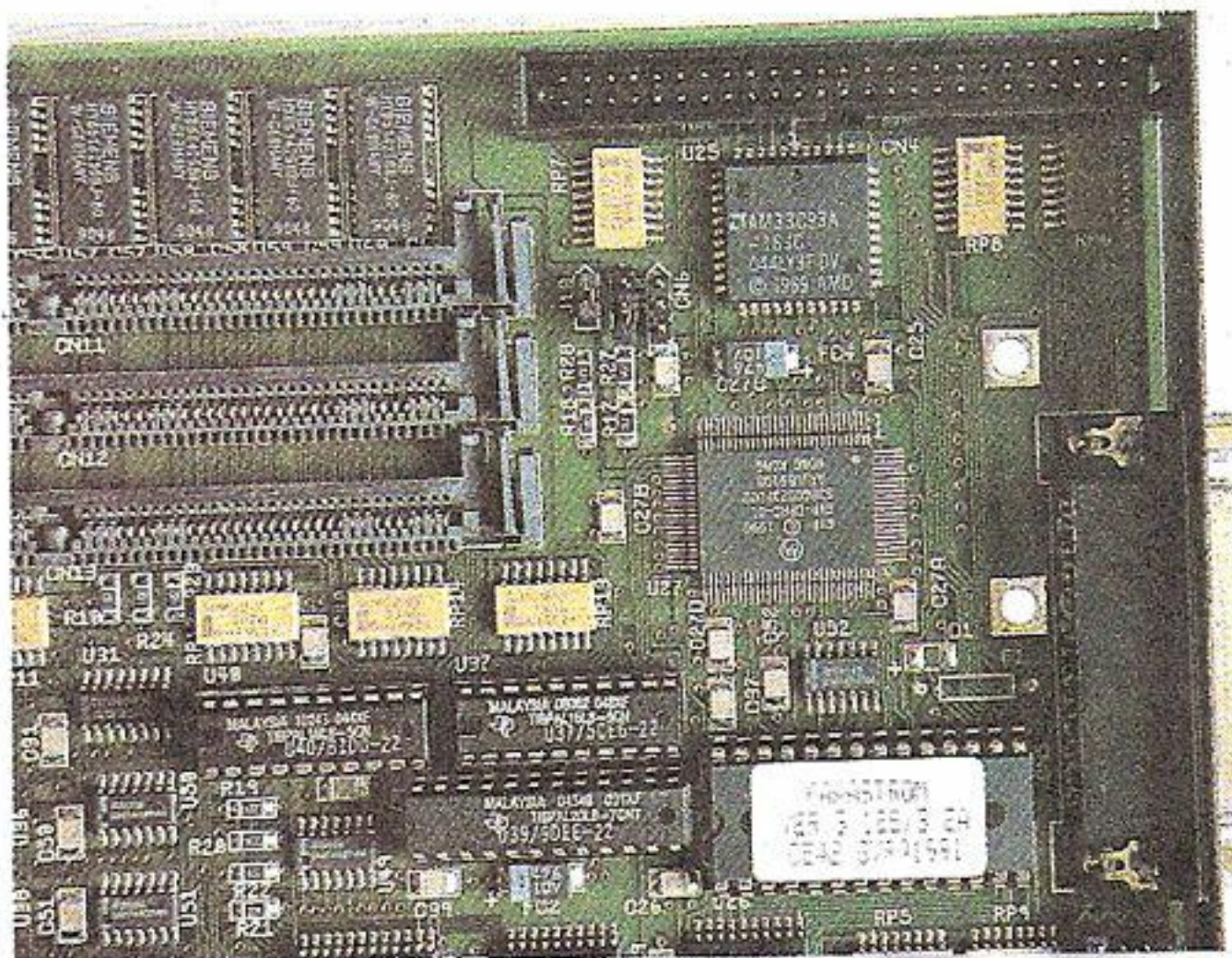
Per installarla, è necessario estrarre dal suo zoccolo la Cpu di Amiga, ovvero il grosso chippone 68000, e inserire al suo posto la basetta, fornita di piedinatura adeguata. Manovra estremamente facile, che richiede unicamente una certa delicatezza per

evitare di danneggiare i piedini dei componenti che si manipolano.

La scheda è composta da un 68000 sulla sua superficie superiore, che differisce dal "collega" originale per la differente velocità di clock: **14.3 Mega Hertz**, praticamente doppia rispetto a quella adottata dai normali Amiga.

La circuiteria sulla faccia inferiore, oltre al necessario per la nuova Cpu, implementa inoltre una cosiddetta **cache ram** di **32 Kb**.

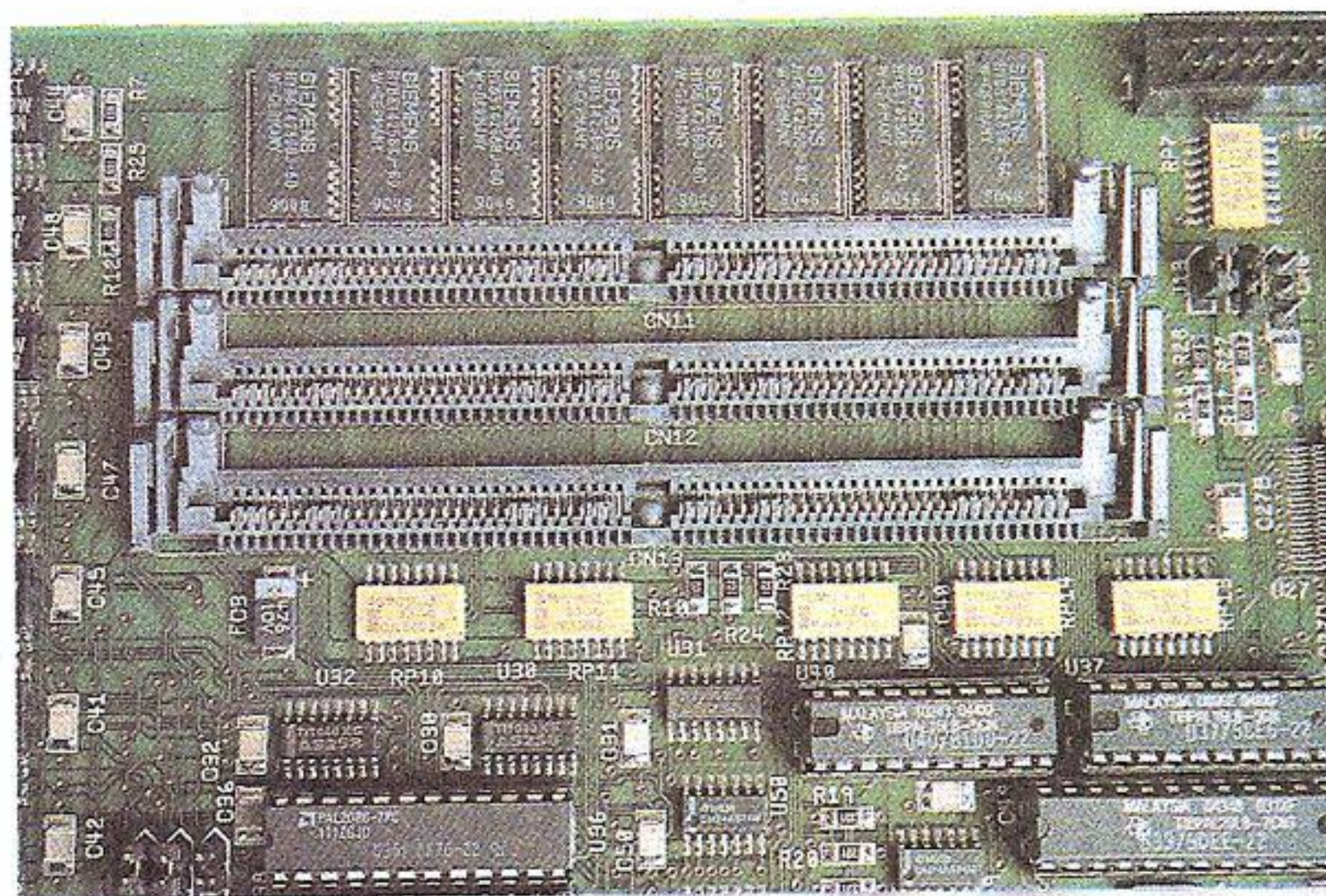
L'incremento della velocità operativa, quindi, è affidata a due componenti dall'effetto diverso, pur se difficilmente quantificabile. La maggiore velocità di clock, infatti, riguarderà solo le operazioni svolte dal 68000, che notoriamente opera in primis sulla cosiddetta **Fast Ram**. In altre parole, trarranno massimo beneficio dalla scheda tutti quei programmi che vengono memorizzati in questo tipo di ram. Tradotto in termini più semplici: praticamente tutti i programmi, in quanto anche quelli che manipolano dati grafici o sonori (che si collocano obbligatoriamente in Chip ram), tendono ad essere memorizzati nella Fast Ram.

Come ovvio, chi possiede un Amiga 500 e vuole adottare questa scheda acceleratrice, per vederne gli effetti dovrà disporre quantomeno di una espansione che vada oltre i 512 Kb di base. L'effetto della *cache ram* contenuta nella scheda sarà poi maggiormente palpabile in tutti quei programmi che svolgono istruzioni o mansioni ripetitive.

I connettori esterno e interno del controller SCSI compreso nella Gvp Impact Serie II, in grado di pilotare fino a 7 periferiche.

Il meccanismo infatti è questo: quando la Cpu accede alla memoria per prelevare un dato, contemporaneamente lo copia nella ram ad alta velocità della scheda. Ad un successivo accesso, controlla prima se il dato è presente nella Cache, e in caso affermativa lo preleva da lì, ad una velocità maggiore che non dovendosi rivolgere alla normale memoria del computer.

L'effetto generale sulla velocità operativa è evidente, ma proprio per quanto appena detto difficilmente quantificabile, poiché non tutti i programmi risponderanno allo stesso modo allo "stimolo" di Adspeed, molto dipenderà da come sono strutturati.

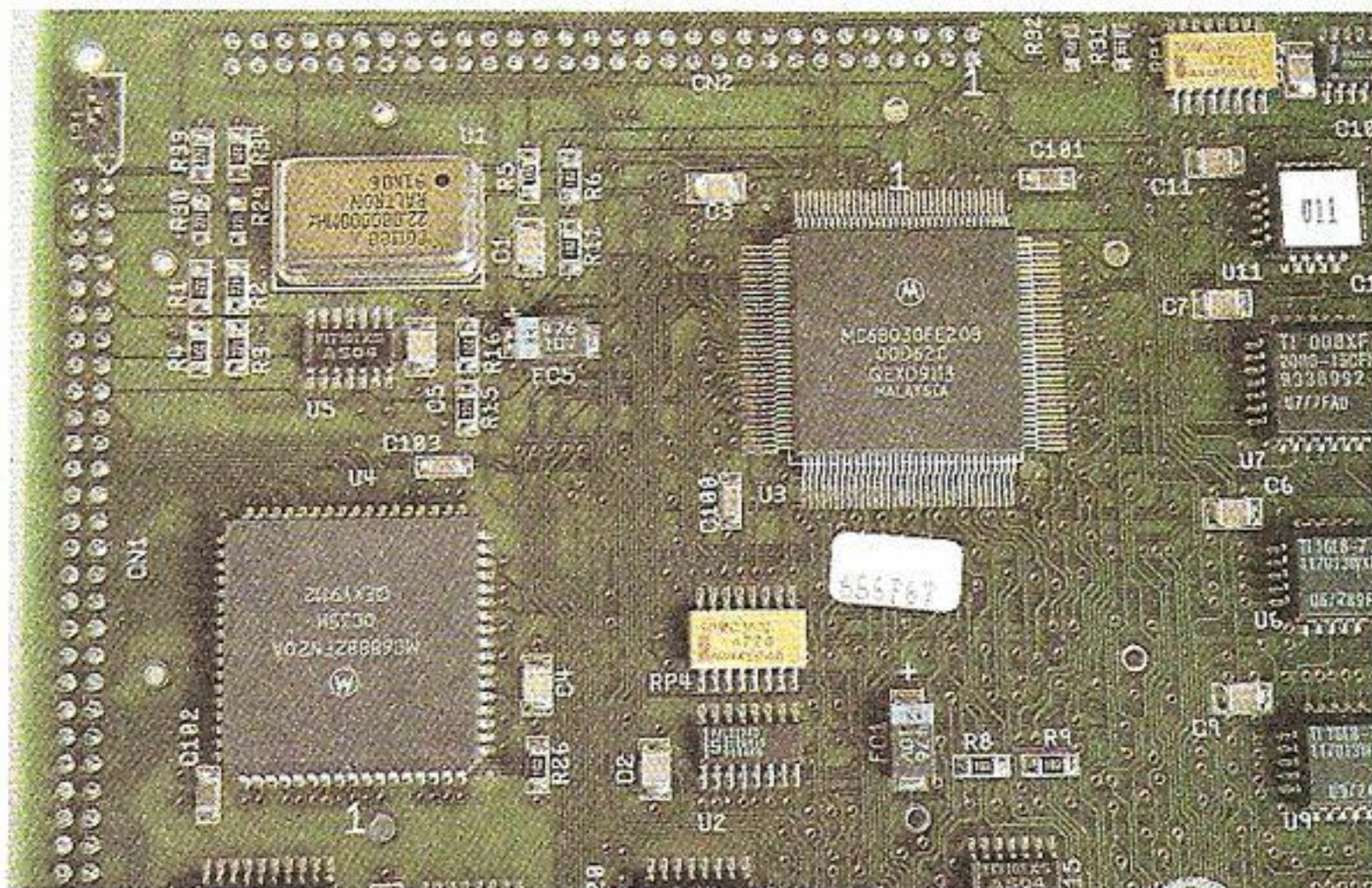
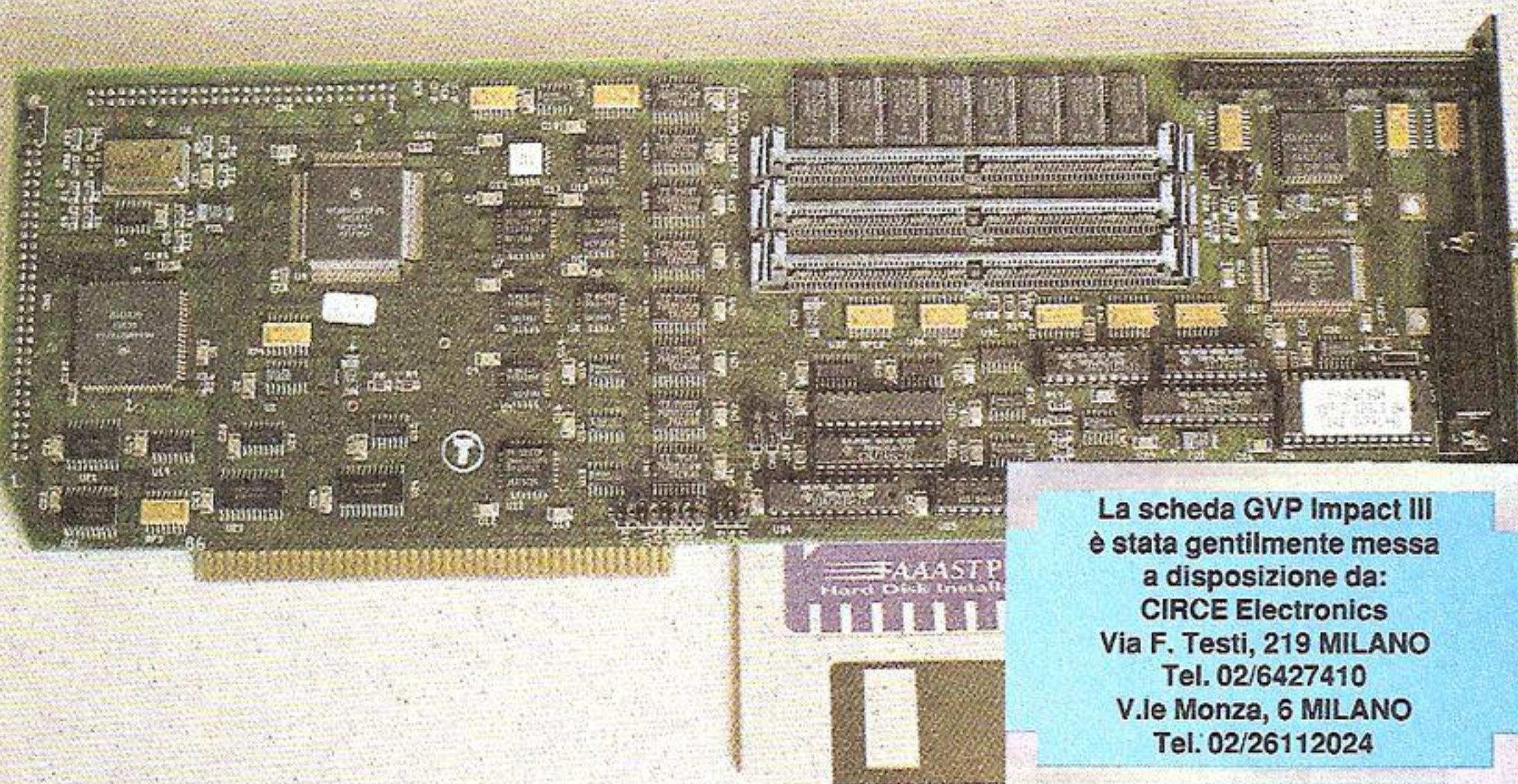
Sulla scheda è anche presente un ponticello per decidere se, all'avviamento del computer, questo deve configurarsi in modo **14.3 MHz** oppure nel tradizionale **7.6 Mhz**. In ogni caso, quale che sia la velocità di start, è poi possibile passare da una all'altra via software, oppure applicando un interruttore che raggiunge l'esterno del computer (non fornito in dotazione). Nel primo caso, basterà clickare da Workbench sull'icona di un programmino presente in un **floppy a corredo** della scheda (eventualmente trasferibile nel proprio disco di start).

La necessità di far "girare" Amiga alla precedente velocità può presentarsi comunque solo con alcuni games che non tollerano accelerazioni. Se proprio non si può fare a meno di questi, e per giunta sono forniti di autostart super protetto, allora si renderà necessario ricorrere alla seconda soluzione hardware, a meno che non si preferisca aprire ogni volta il computer e modificare lo stato del jumper sulla scheda.

La Ram aggiuntiva montata sulla scheda GVP Impact (1 oppure 4 Mb), e gli slot Simm che possono contenerne altri 12 Megabyte.

Gvp Impact Series II

Con le nuove schede **Impact** della **GVP**, usare il termine *acceleratrici* è sicuramente riduttivo.

Ci si trova infatti di fronte a funzioni multiple, in grado di trasformare letteralmente un Amiga 2000 in qualcosa di molto, molto diverso, che in quanto a velocità di elaborazione e accesso ai dati su hard disk non ha nulla da invidiare ad un A-3000. Molto semplicemente (si fa per dire), le GVP Impact trasformano il 2000 in un computer a **32 bit** grazie al microprocessore **68030**, cui è affiancata una unità di calcolo **68882**, il tutto proposto in **tre differenti versioni** in rapporto alla velocità di clock implementata: **22** Mhz, **33** MHz, oppure **50** Mhz.

Il costo, come intuibile, varia molto a seconda del modello, come del resto le prestazioni, che possono teoricamente superare quelle di un A-3000.

Il "cuore pulsante" della scheda GVP: microprocessore 68030, coprocessore matematico 68882 e oscillatore a 22, 33 oppure 50 Mhz.

La scheda GVP Impact III
è stata gentilmente messa
a disposizione da:
CIRCE Electronics
Via F. Testi, 219 MILANO
Tel. 02/6427410
V.le Monza, 6 MILANO
Tel. 02/26112024

La scheda Impact Series II della GVP, un mostro di potenza per Amiga 2000.

Considerato il costo comunque non irrisorio, abbiamo provato il modello più accessibile sotto questo aspetto, quello gestito dall'oscillatore a **22 MHz**.

Intanto c'è da dire che la scheda svolge in pratica mansioni che normalmente necessitano di almeno due accessori hardware.

Alla sezione acceleratrice vera e propria, si affianca infatti un controller SCSI di velocità superlativa, in grado di gestire fino a **7 periferiche** aderenti a questo standard, collegabili tanto internamente che esternamente grazie a due connettori adatti allo scopo.

Non manca naturalmente una ROM per l'**autoboot** del sistema da hard drive.

Adoperando la GVP, inoltre, non è più necessario disporre di schede di espansione, che anzi sono sconsigliate.

La 68030, infatti, monta già a bordo **1 Mega di ram** nella versione 22 MHz e **4 Mega** nelle superiori, ad acceso veloce (60 ns) e a 32 bit.

A questa configurazione di base possono poi essere aggiunti moduli SIMM fino ad un totale (teorico) di 16 Mega byte, interamente gestiti dalla scheda.

La configurazione è regolabile con molta semplicità attraverso tre ponticelli, la cui apertura/chiusura è deducibile da chiari schemi illustrati nel manuale in dotazione. Considerate le sue caratteristiche multiple, la scheda GVP Impact consente in pratica di occupare **un solo slot** di Amiga 2000, quello denominato *Co-Processor Expansion* (il più vicino ai drive interni), per prestazioni che, anche nella versioni 22 MHz, possono essere sintetizzate da un aggettivo: **mostruose**.

La confezione avuta in prova comprendeva solo la scheda, senza la meccanica dell'hard disk, la presenza della quale fa ovviamente innalzare il normale prezzo di vendita.

Del resto, sotto l'aspetto transfer rate da/verso l'hard disk, già le GVP Impact della generazione immediatamente precedende (che non implementavano lo SCSI) erano considerate tra le più veloci esistenti: non ci si può dunque aspettare sorprese, se non in senso positivo. Se ne riparlerà, comunque, in un prossimo appuntamento.

Una valutazione obiettiva della sezione acceleratrice, non può comunque che essere entusiastica. Considerando la versione a 22 MHz, si pensi a una differenza che può essere semplificata da due prove sperimentali.

La prima, non può che riguardare il settore ove una unità di calcolo può esprimere tutta la sua potenza: il cosiddetto **rendering**, ovvero la riproduzione tridimensionale con effetti di luce/ombra adeguata, di una immagine prima bidimensionale. Per il test si è adoperato il programma **Draw 4D**, recensito qualche numero fa, e globalmente uno dei più veloci del settore.

Si è scelta un'immagine non troppo complessa, ma probante: una titolazione in 3D a tutto schermo.

Sorvolando sulla nuova fluidità e velocità del movimento tridimensionale lungo gli assi spaziali, il cronometro (i vecchi metodi fanno talvolta poesia...) ha sancito questi risultati: il *rendering solido* dell'immagine è stato completato **senza** la scheda GVP in **3 minuti e 5 secondi**.

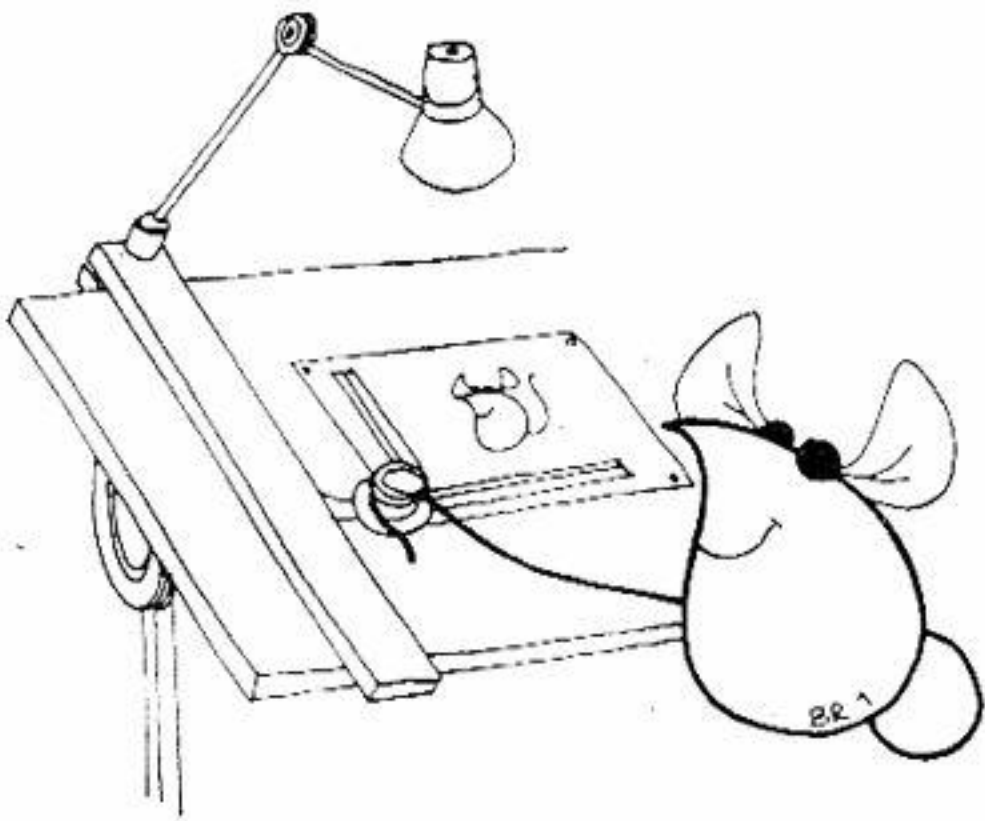
Non male, direte, anche se l'idea di creare un'animazione di 60 frame a questo ritmo non fa certo sorridere. Attivata la scheda, la stessa identica operazione è stata eseguita in... indovinate un po'? **25 secondi!** Il pensiero di cosa è capace di fare la stessa scheda in versione **50 Mhz** fa venire i brividi.

Senza andare sul complicato, si è anche deciso di provare una banale iterazione sfruttando il notoriamente lumacone interprete **AmigaBasic**. Un ciclo For...Next da 1 a 100000 che normalmente impiega 45 secondi, con la Gvp viene ultimato in 7 secondi.

Cos'altro aggiungere?

Beh, qualcosa ci sarebbe: la constatazione che ci si trova al cospetto di un prodotto di serie A, non solo nelle prestazioni, ma anche nel resto: **manuale molto chiaro** ed esplicativo, software in dotazione di tutto rispetto.

Quest'ultimo, in particolare, è riservato in massima parte alla gestione e preparazione di un hard disk (senza dimenticare l'interazione con eventuali schede **Janus** di emulazione Ibm).

Non mancano un test completo della memoria ram, ed un interessantissimo **GvpInfo**, in grado di fornire informazioni molto dettagliate sulle principali risorse del sistema.

Al tutto, si aggiunge poi la possibilità di *rebootare* il sistema in normale modo 68000, nel caso si verificassero incompatibilità con qualche programma.

Questa feature può essere attivata tanto via software che via hardware, modificando in quest'ultimo caso lo stato di un jumper.

Resta da vedere se, una volta cominciato a smanettare su Amiga con la sua nuova veste 68030 a 22 (o più) Mega Hertz, verrà mai la voglia di tornare a quello... scansafatiche di un 68000!

Il software di installazione per hard disk fornito assieme alla scheda GVP Impact.

✔ Amiga
❏ Ms - Dos
❏ Recensioni
❏ Hardware
❏ Software
✔ Applicazioni
✔ Programmazione
❏ Dos
❏ Pascal
❏ C
✔ Basic
❏ Assembly

❏ Principianti ❏ Esperti ✔ Tutti ✔ Help

Il brevissimo programma in AmigaBasic affronta e risolve brillantemente il principale problema della grafica 3-D: la rotazione dei solidi in prospettiva

Solidi rotanti in prospettiva

< di Luca Cassioli>

< Esaminare C. C. n. 42 e n. 80 > < Si lancia anche una sfida ai lettori! >

Il programma di queste pagine risponde alla sfida lanciata sul lontano numero **80** in cui fu pubblicato l'articolo "**...ed è subito prospettiva**" che rappresentava, appunto, un'introduzione allo studio della prospettiva.

Il listato del n. 80 (implementato in AmigaBasic, Gw-Basic e Turbo Pascal) si limitava a visualizzare due quadrati sfalsati l'uno rispetto all'altro, rispettando le "regole" della prospettiva, come se fossero le due facce opposte di un cubo. Il programma qui sviluppato vuole essere un miglioramento dello stesso, in quanto fa uso di un algoritmo leggermente più complesso per il calcolo delle coordinate.

Inoltre permette di far ruotare rispetto agli assi **X**, **Y** e **Z** il solido definito dai valori inseriti all'inizio. Si fa un uso particolare, e abbastanza complesso, di **variabili indicizzate**, rese necessarie per evitare di ripetere più volte, nel corso del programma, routine simili tra loro per realizzare l'elaborazione di dati poco diversi gli uni dagli altri.

L'algoritmo del sottoprogramma che si occupa di far ruotare le coordinate dei singoli vertici è stato tratto dal programma per Commodore 64, di **Carlo Boni**, apparso sul numero **42** di CCC alle pagine da 31 a 34; allo stesso programma ci si è "ispirati" per quanto riguarda alcuni assegnamenti di variabili.

Entrambi i programmi, infatti, disegnano i solidi tracciando i segmenti ottenuti unendo le coppie di vertici indicate dalle variabili **SP1()** e **SP2()**. Tuttavia il programma di queste pagine contiene alcune modifiche essenziali che lo differenziano in buona parte dall'altro, come vedremo nella sezione in cui viene descritto il funzionamento punto per punto.

B'
A α B
A'
O

Figura 1

Figura 2

Utilizzo del programma

Per semplificare al massimo le cose non è presente una routine per l'in-

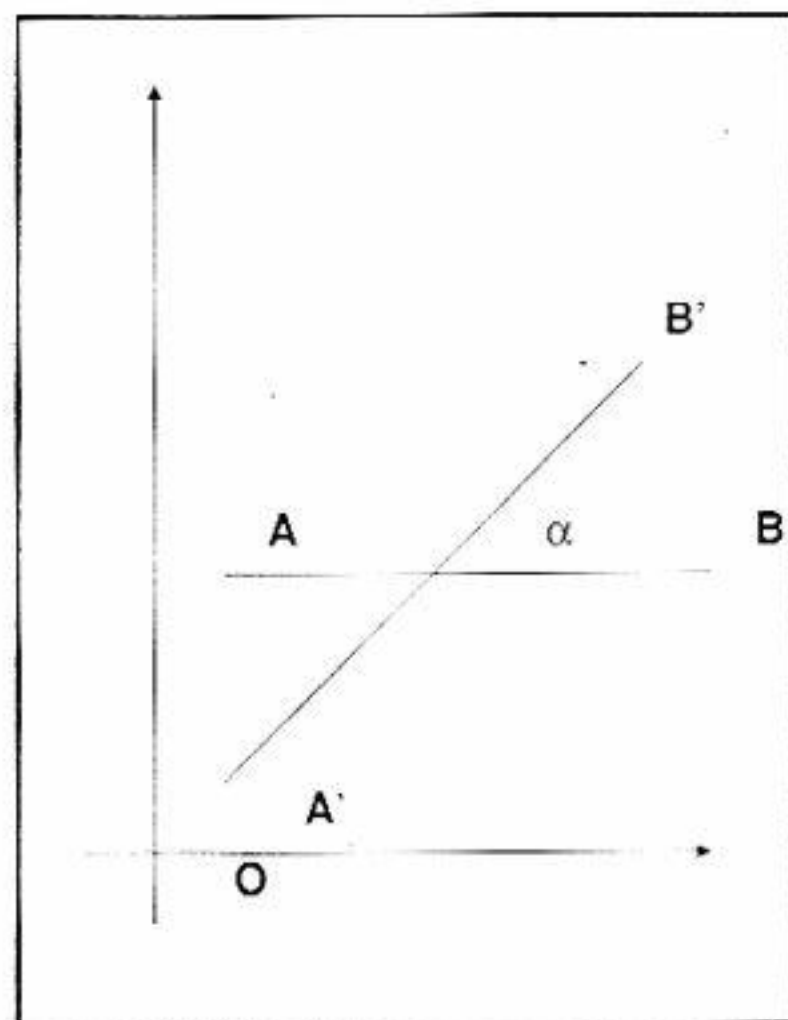
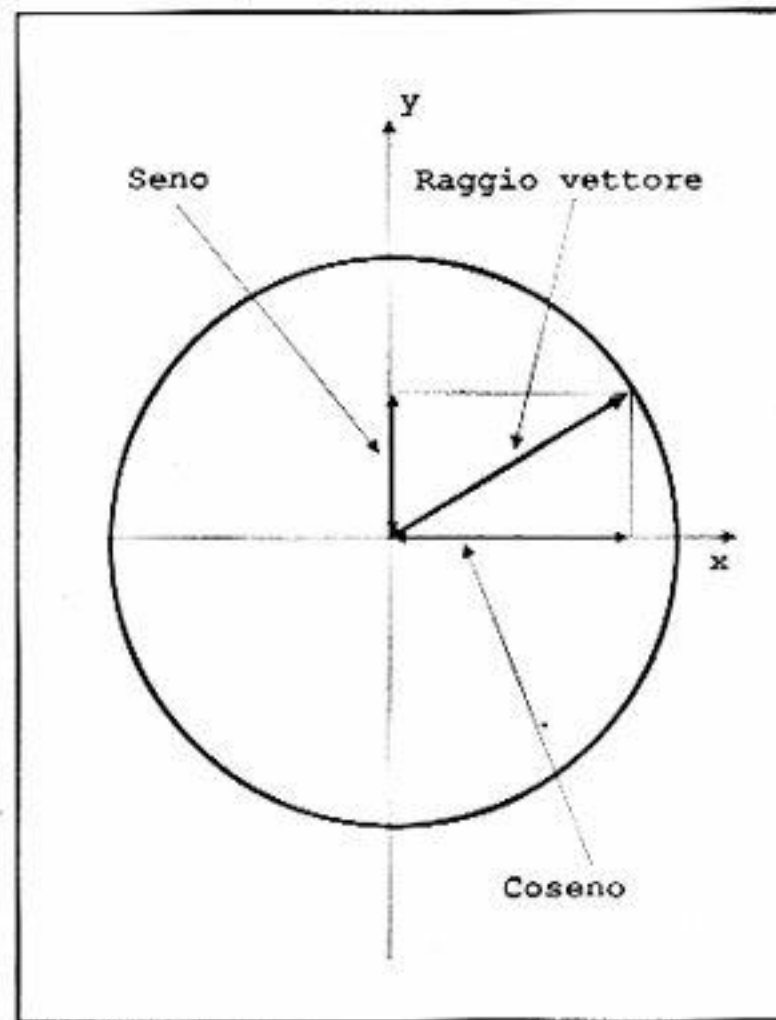
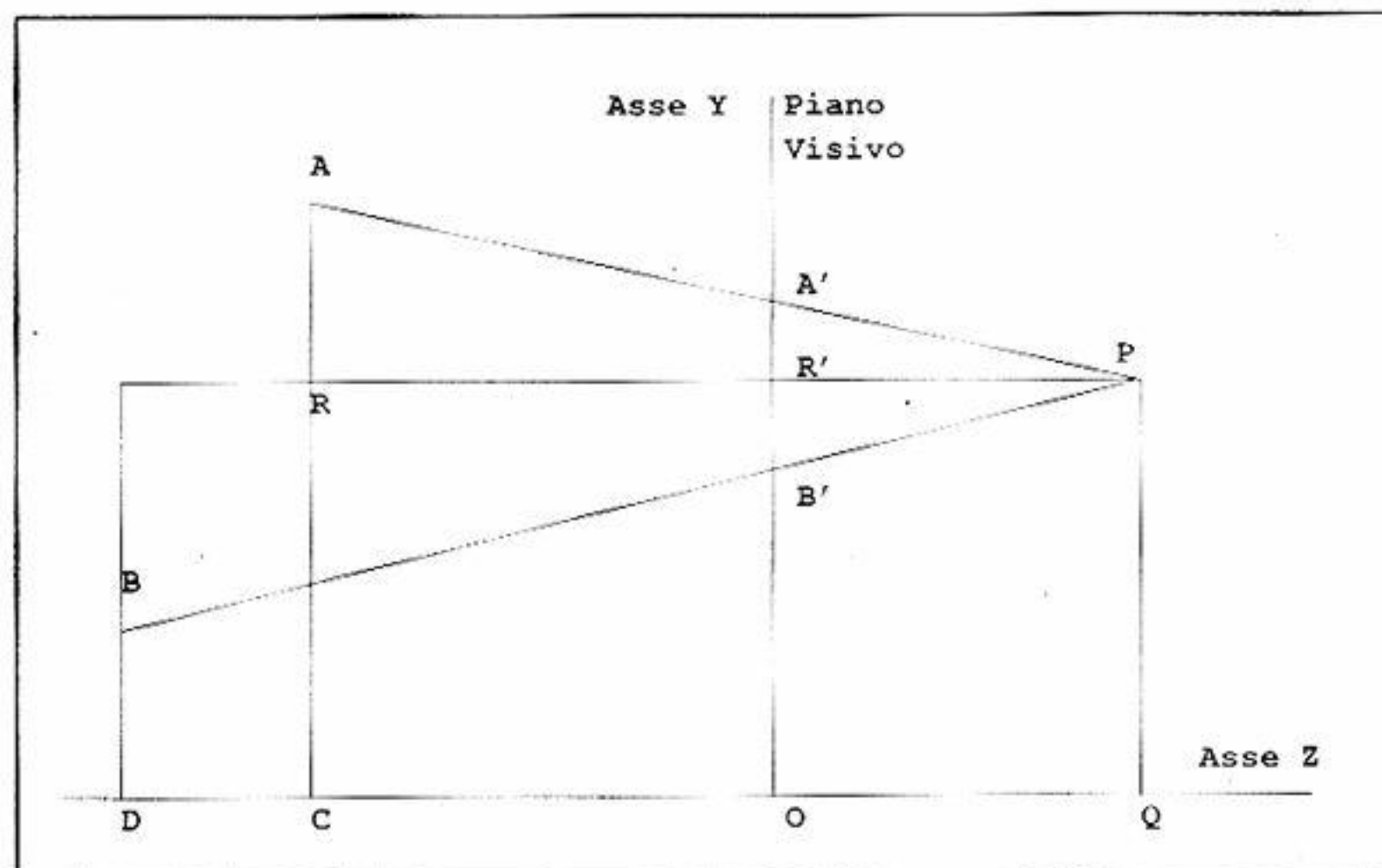



Figura 3a
A e B sono i punti nello spazio di cui vogliamo calcolare le coordinate Y sul piano visivo, il che risulta semplice, per il principio della similitudine dei triangoli, conoscendo AC (coordinata Y di A), BD (coordinata Y di B), PQ (coordinata Y di P) e CQ (somma delle coordinate Z dei punti A e P prese in valore assoluto) oppure DQ (coordinate Z dei punti B e P prese in valore assoluto).

serimento dei parametri necessari, assegnati all'inizio dallo stesso programma.

Tali parametri determinano la visualizzazione di un cubo che misura un lato di **80 pixel**, al centro del quale sarà visibile il *punto di fuga* (**pdf**) che ha le stesse coordinate X e Y del *punto di vista* (**pdv**), ma **diversa** coordinata **Z**; infatti, il punto di fuga possiede una distanza teoricamente infinita dall'osservatore, mentre la coordinata Z del punto di vista indica la posizione, nello spazio, dell'osservatore stesso. All'avvio del programma, saranno disponibili numerose opzioni, tutte attivabili da tastiera alla semplice pressione di un tasto. Eccone l'elenco:

Figura 3b
A e B sono gli stessi punti della figura 3.a, ma questa volta sono visti dall'alto. Il procedimento per trovare le loro coordinate X è analogo al precedente. Da notare che in entrambe le figure i punti, per semplicità, occupano solamente i due semipiani positivi YZ e XZ.

- Tasti **G**, **H**: rotaz. asse **Z** nei due sensi.
- Tasti **Y**, **B**: idem per l'asse **Y**
- Tasti **U**, **N**: idem per l'asse **X**
- Tasti **A**, **S**: sposta a sinistra e a destra.
- Tasti **W**, **Z**: idem in alto e in basso.
- Tasti **-**, **+**: avvicina e allontana solido.
- Tasti **R**, **C**: alza e abbassa il pdv.
- Tasti **D**, **F**: pdv a destra e a sinistra.
- Tasti **T**, **V**: allontana e avvicina il pdv.
- Tasto **1**: termina il programma.

La **figura 4** mostra la convenzione usata nel programma per identificare i tre assi.

Il programma

Inizialmente viene aperto un nuovo schermo in bassa risoluzione e, all'interno di esso, una finestra.

Ciò non viene fatto per abbellire il programma, ma per evitare la deformazione dei solidi visualizzati, in quanto con la risoluzione 640 x 200 i pixel risultano essere larghi circa la metà della loro altezza; di conseguenza anche le misure orizzontali sono dimezzate.

In pratica un segmento lungo 50 pixel, ad esempio, sarebbe lungo 5 centimetri in orizzontale e 10 in verticale.

Subito dopo vengono assegnate alcune costanti. L'array COPYS() contiene i valori **+1** e **-1**, mentre l'array COPYX() contiene, a coppie, i valori **1**, **2** e **3**, che corrispondono, in una successiva elaborazione, rispettivamente alle coordinate **X**, **Y** e **Z**.

Tale corrispondenza ritorna frequentemente nel programma, dato il largo uso di matrici in cui uno degli indici è proprio il "tipo" di coordinata.

In particolare, queste due costanti vengono utilizzate dalle routine che si occupano di traslare sia il punto di vista che il solido.

Delle altre variabili utilizzate, la variabile **VAR (C, V)** contiene tutte le coordinate di tutti i vertici del solido; **C** può assumere i valori da **1** a **3**, con la corrispondenza vista prima, mentre **V** può assumere valori da 1 al numero di vertici.

In pratica, la variabile VAR (1, 5) conterrà la coordinata X del vertice 5, VAR (3, 12) conterrà la variabile Z del vertice 12 e così via.

In questo modo è più semplice traslare o ruotare insieme tutti i vertici tramite un

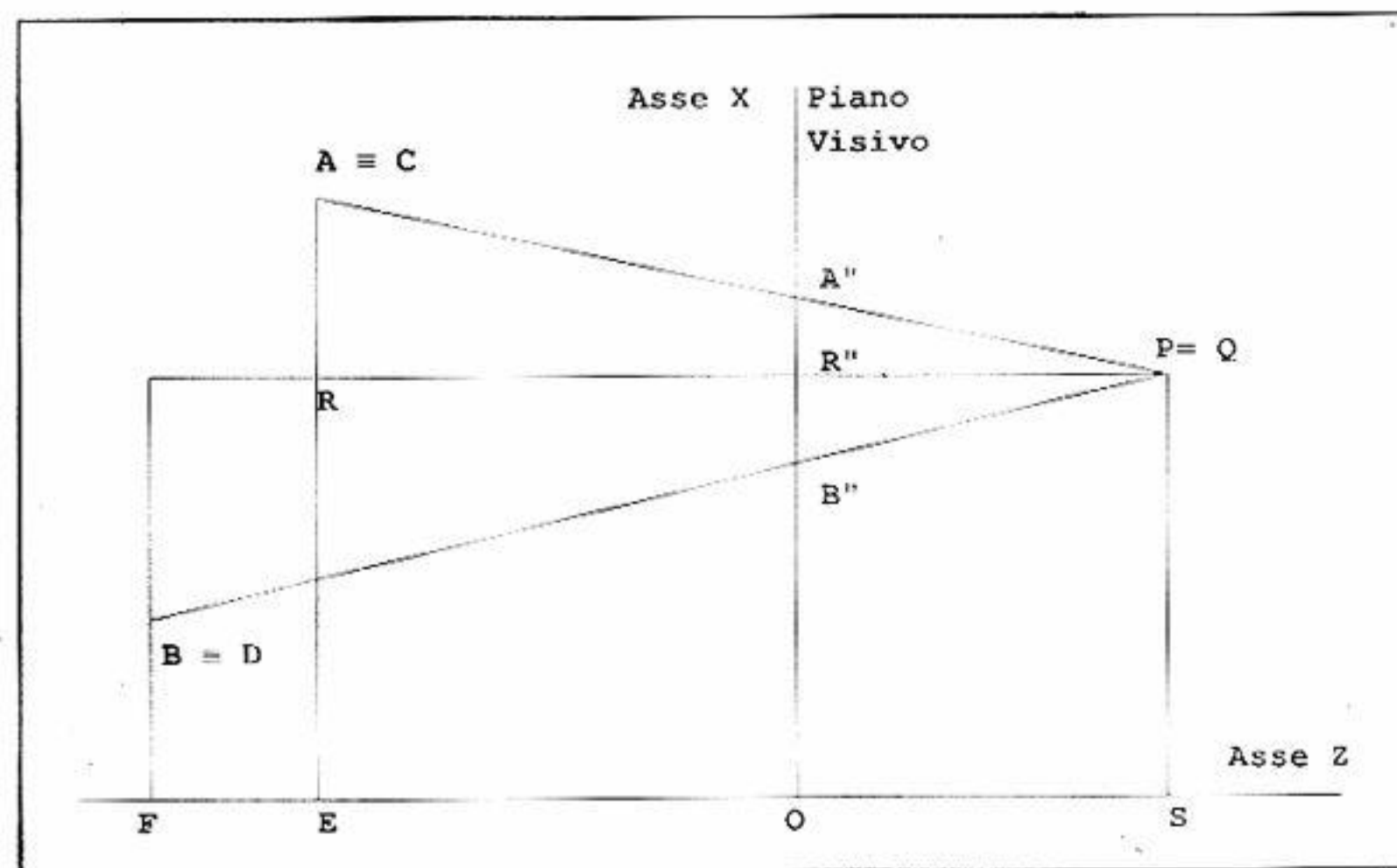
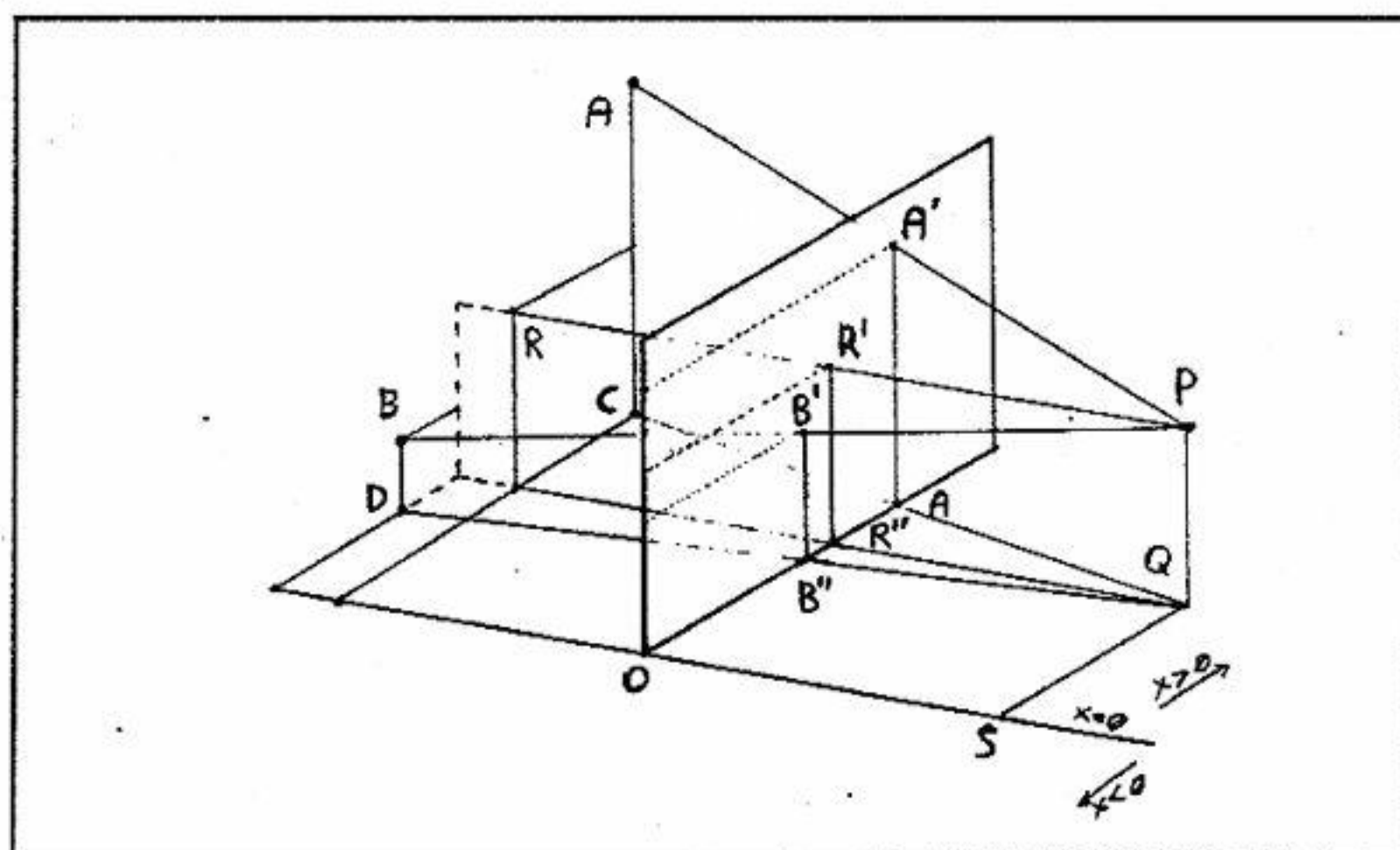



Figura 3c

ciclo FOR...NEXT e l'utilizzo di altre matrici come indici di questa. La routine di traslazione del solido identificata, ad esempio, dal numero di linea 1, incrementa (COPYS(TT)=1) o decrementa (**COPYS(TT)=-1**) della quantità **MOV** la coordinata X (**COPYX(TT)=1**), Y (**COPYX(TT)=2**) oppure Z (**COPYX(TT)=3**) di ogni vertice (**FOR tr = 1 TO nv: ... :NEXT**).

I valori variano in base a quello contenuto nella variabile **TT**, risultato dell'elaborazione della routine al termine del listato, che si occupa di esaminare il tasto premuto. La routine di traslazione del punto di vista funziona in modo analogo, con la differenza che la quantità tolta o aggiunta è sempre **10**.

Le coordinate X, Y e Z del punto di vista sono contenute nelle variabili PV(1), PV(2) e *PV(3)*.

Come ruota

La routine di rotazione, invece, è ovviamente più complessa. Dopo aver assegnato alla variabile **AN** l'angolo di rotazione in radianti, tramite il solito ciclo FOR...NEXT vengono elaborati tutti i vertici, ma solo dopo aver calcolato il baricentro del solido per evitare che questo, in seguito alla rotazione, fuoriesca dallo schermo. La routine che si occupa del calcolo è denominata **Centro**, e segue la routine **Maggiore**, in cui vengono calcolate le coordinate massime e minime di ogni solido.

Infatti le coordinate X, Y e Z del baricentro non corrispondono alle medie delle coordinate X, Y e Z di ogni vertice, come invece viene calcolato nel succitato programma di Carlo Boni.

Ognuna di esse, invece, corrisponde alla media tra la **massima** e la **minima** coordinata corrispondente.

Esempio pratico:

```
Xmax = 300: Xmin = 20
Ymax = 180: Ymin = 40
Zmax = 200: Zmin = -100
Xbar = (300 + 20) / 2 = 160
Ybar = (180 + 40) / 2 = 110
Zbar = (200 + (-100)) / 2 = 50
```

La routine che si occupa effettivamente della rotazione è invece tratta dal vecchio programma (CCC n. 42), ed è stata qui inserita in un sottoprogramma in modo che possa essere utilizzata anche separatamente, ricordandosi però di inserire l'angolo di rotazione in **radianti**.

Chi non ama la trigonometria può passare direttamente al paragrafo successivo, in cui viene spiegato in modo approfondito l'algoritmo.

L'algoritmo principale

Nel programma di Carlo Boni venivano utilizzate tre routine simili tra loro per ottenere lo stesso risultato per le tre diverse coordinate.

Grazie al metodo delle matrici è invece possibile utilizzare un'unica routine per le tre funzioni. Vediamo come.

Immaginiamo di avere un segmento **AB** sul piano **XY**, parallelo all'asse **X** (**figura 1**), e di volerlo ruotare di 30 gradi in senso antiorario. Per fare ciò, occorre prendere in considerazione le funzioni trigonometriche **seno** e **coseno**.

Ricordiamo, innanzitutto, che la Circonferenza Trigonometrica è una circonferenza di raggio unitario con centro nell'origine di un sistema di assi cartesiani;

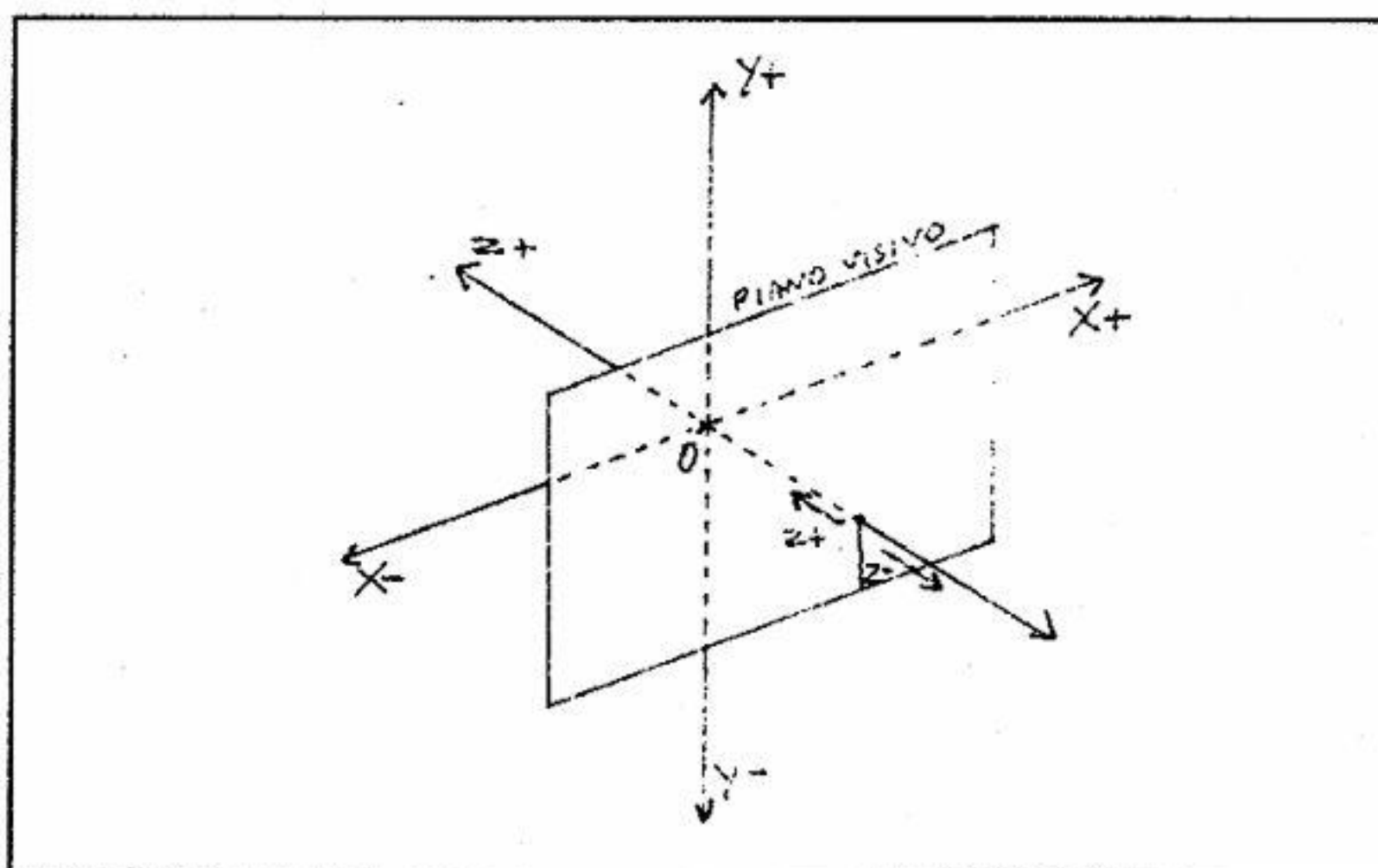



Figura 4

da qui ricaviamo la definizione secondo cui il seno di un angolo α equivale alla proiezione sull'asse delle Y del raggio della circonferenza, il quale raggio forma un angolo α con l'asse delle X, mentre il coseno è la proiezione dello stesso segmento sull'asse X.

In altre parole, il coseno e il seno di un angolo, sono, rispettivamente, le coordinate X e Y del punto della circonferenza goniometrica che, unito con l'origine, forma un segmento formante un angolo α con l'asse delle X (**figura 2**), ed è proprio sotto questo punto di vista che interessano le due funzioni.

Per determinare le nuove coordinate del punto **A** del segmento (fig. 1) dopo la rotazione, occorreranno l'angolo di rotazione e la distanza del punto dal centro di rotazione, ed è per questo che il sottoprogramma richiede anche le coordinate del centro di rotazione.

Tale distanza serve perché le funzioni seno e coseno danno sempre risultati compresi tra **-1** e **+1**, mentre i punti da ruotare potranno essere più distanti di una unità dal centro di rotazione. La distanza, quindi, viene moltiplicata per il risultato della funzione seno o coseno, per trovare rispettivamente le coordinate Y e X del punto ruotato; in seguito ai due risultati vengono aggiunti i valori delle corrispondenti coordinate del centro di rotazione, ed il gioco è fatto.

Un'ultima nota riguarda l'angolo di cui viene calcolato il seno o coseno, che non è sempre quello inserito all'inizio, ma la somma di questo con l'angolo di cui ogni punto è già stato ruotato, il cui valore è contenuto in WK.

Quest'ultimo equivale all'**arcotangente** del rapporto tra coordinata Y e coordinata X del punto da ruotare, sia per il teorema dei triangoli rettangoli (che recita: *in un triangolo rettangolo la tangente di un angolo acuto è uguale al rapporto tra cateto opposto e cateto adiacente*) sia perché l'arcotangente è l'angolo a cui corrisponde un valore N della tangente.

Infatti il punto avente coordinate X, Y può essere considerato un vertice del triangolo rettangolo i cui altri vertici hanno coordinate (X, 0).

Per quanto riguarda il calcolo della distanza punto-da-ruotare / centro-di-rotazione, esso è effettuato tramite la formula matematica:

$$d = sqr\ ((y1 - y2)^2 + (x1 - x2)^2)$$

...di cui non staremo a fare la dimostrazione.

Chi fosse interessato, non dovrà che consultare un qualsiasi testo di trigonometria, o arrivarci per intuito tenendo in considerazione il **teorema di Pitagora**.

Il procedimento, apparentemente valido soltanto per rotazioni intorno all'asse Z, è in realtà adatto anche a tutti gli altri punti; basta infatti fornire le coordinate X e Z per effettuare una rotazione rispetto all'asse Y, ed Y e Z per ruotare intorno a X; in seguito penserà la routine **AssonPros** ad elaborare in modo opportuno le varie coordinate ottenute.

Per tale motivo è possibile usare lo stesso sottoprogramma per le tre coordinate.

Assonometria e prospettiva

La routine **AssonPros** elabora ulteriormente le coordinate, delle quali, altrimenti, potrebbero essere utilizzate solo le X e Y, da cui risulterebbe una visualizzazione del solido in assonometria.

Le formule utilizzate sono diverse da quelle comparse nel programma pubblicate sul n. 80, perché queste considerano anche l'eventualità di spostare il punto di vista; con le altre, invece, questo era considerato sempre giacente sul piano ZX, "per terra", per intenderci.

Per la formulazione delle relazioni matematiche si è seguito un procedimento analogo, sempre basato sulla similitudine di triangoli, illustrato in **figura 3**.

Chiude il programma la subroutine **Tasto**, cui accennavamo prima, che, tramite un ciclo FOR...NEXT, confronta il tasto premuto con tutti i caratteri della costante stringa dichiarata all'inizio, contenente i caratteri consentiti, e restituisce un valore compreso tra **1** e **19**, gestito dalla successiva routine **Getkey**.

Il programma non prevede la possibilità di inserire manualmente parametri, nè di salvarli eventualmente su disco, ma implementare tali opzioni non dovrebbe essere difficile.

Un'ultima nota: per provare il programma, si possono inserire anche i dati riportati a pagina 30 del numero 42 di CCC, che venivano forniti come esempio per il programma di Carlo Boni, e che si adattano anche a questo per le somiglianze spiegate.

La casetta risulterà un po' troppo grande, ma basterà premere un paio di tasti per rimpicciolirla e poterci poi giocare come si vuole...

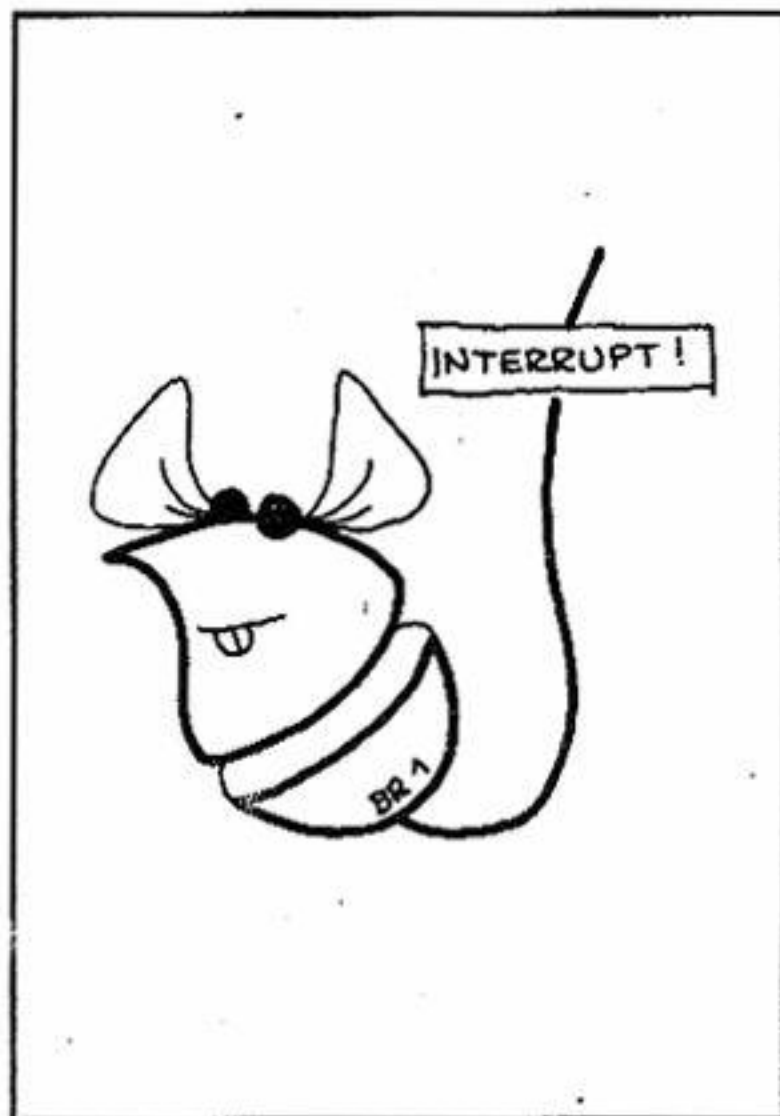
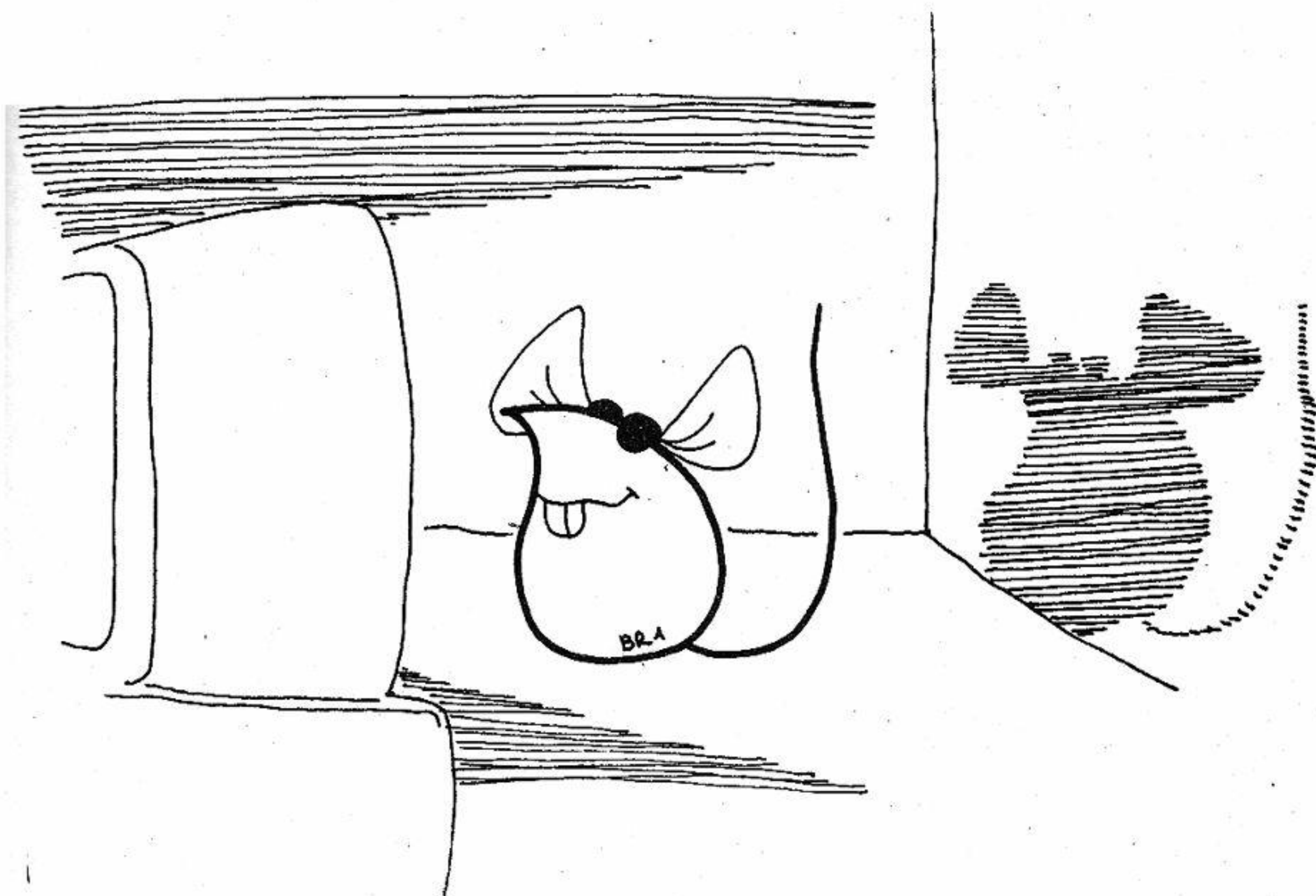
Una sfida ai lettori

La soluzione del problema, e la velocità relativamente elevata possibile perfino in AmigaBasic, può esser considerata come base di partenza per l'implementazione dello stesso algoritmo in **Amos**, **C** oppure **Pascal**.

"Attorno" alla procedura potrebbe essere sviluppata, parallelamente, la possibilità di importare, da un file ASCII, le coordinate dei vertici di una qualsiasi figura da far ruotare, in seguito, dopo averla incorporata nel programma.

Il progetto è affascinante, anche se complesso. Del resto il programma che vedete in queste pagine ha origine, nientemeno, che nel n. 42, quando di Amiga ancora non si parlava.

Oggi, con i nuovi e potenti linguaggi dell'ultima generazione, forse sarà necessario attendere un aggiornamento in tempi più brevi.

La sfida è lanciata. Se l'accettate, comunque, fatevi sentire prima della fine di questo secolo...

```
'Rotazioni in prospettiva V 1.0
'       By Luca Cassioli
'    S.Angelo Romano - ROMA
'
 SCREEN 1,320,256,2,1
 WINDOW 1,"Prospettiva",(0,0)-(310,240),2,1
 copyx(1)=1:copyx(2)=1:copyx(3)=2:copyx(4)=2:copyx(5)=3:copyx(6)=3
 copys(1)=+1:copys(2)=-1:copys(3)=+1:copys(4)=-1:copys(5)=+1:copys(6)=-1
 pigreco=3.14159
 rot(1)=1:rot2(1)=2:rot(2)=1:rot2(2)=3:rot(3)=2:rot2(3)=3
 ctr$="sawz+-fdrctvghybun1"
 DIM prox(100):DIM proy(100)
Inserimento:
 pv(1)=160:pv(2)=100:pv(3)=-100:nv=8:sp=12:mov=30:ang=22.5
 DIM var(3,nv),sp1(sp),sp2(sp)
 var(1,1)=120:var(2,1)=60:var(3,1)=0:var(1,2)=200:var(2,2)=60:var(3,2)=0
 var(1,3)=200:var(2,3)=140:var(3,3)=0:var(1,4)=120:var(2,4)=140:var(3,4)=0
 var(1,5)=120:var(2,5)=60:var(3,5)=80:var(1,6)=200:var(2,6)=60:var(3,6)=80
 var(1,7)=200:var(2,7)=140:var(3,7)=80:var(1,8)=120:var(2,8)=140:var(3,8)=80
```

```
 sp1(1)=1:sp2(1)=2:sp1(2)-2:sp2(2)=3:sp1(3)=3:sp2(3)=4:sp1(4)=4:sp2(4)=1
 sp1(5)=5:sp2(5)=6:sp1(6)=6:sp2(6)=7:sp1(7)=7:sp2(7)=8:sp1(8)=8:sp2(8)=5
 sp1(9)=1:sp2(9)=5:sp1(10)-2:sp2(10)=6:sp1(11)=3:sp2(11)=7
 sp1(12)=4:sp2(12)=8
 GOSUB Disegna
Getkey:
 a$=INKEY$:IF a$="" THEN Getkey
 GOSUB Tasto:
 IF tt=99 THEN Getkey
 IF tt>12 THEN sig=SGN(INT(tt/2)-(tt/2)+.1)  ' sig=senso di rotazione
 ON tt GOTO 1,1,1,1,1,1,2,2,2,2,2,2,Calc,Calc,Calc,Calc,Calc,Calc,fine
1 xyz=copyx(tt):spost=copys(tt)
  FOR tr=1 TO nv:var(xyz,tr)=var(xyz,tr)+(spost*mov):NEXT
  GOSUB Disegna:GOTO Getkey
2 PSET (pv(1),200-pv(2)):pv(copyx(tt-6))=pv(copyx(tt-6))+copys(tt-6)*10
  GOSUB Disegna:GOTO Getkey
Calc:
    an=ang*sig:an=an*pigreco/180
    GOSUB Maggiore
    FOR a=1 TO nv:au1=var(rot(copyx(tt-12)),a):au2=var(rot2(copyx(tt-12)),a)
    ruota.x an,au1,au2,c1(copyx(tt-12)),c2(copyx(tt-12))
    var(rot(copyx(tt-12)),a)=au1:var(rot2(copyx(tt-12)),a)-au2:NEXT
    GOSUB Disegna
    GOTO Getkey
 IF a$="1"THEN CLS:END
 GOTO Getkey
Disegna:
 CLS
 GOSUB AssonPros:FOR di=1 TO sp
 LINE (prox(sp1(di)),200-proy(sp1(di)))-(prox(sp2(di)),200-proy(sp2(di))),1
 NEXT:PSET (pv(1),200-pv(2)),1:GOSUB Maggiore:RETURN
AssonPros:
 FOR a=1 TO nv
 prox(a)=(((var(1,a)-pv(1))*ABS(pv(3)))/(var(3,a)+ABS(pv(3))))+pv(1)
 proy(a)=(((var(2,a)-pv(2))*ABS(pv(3)))/(var(3,a)+ABS(pv(3))))+pv(2)
 NEXT:RETURN
'Immettere: Angolo rotazione in radianti
'Coordinate Y e Z del punto
'Coordinate Y e Z centro rotazione
'
SUB ruota.x(rk,y,z,yc,zc) STATIC
 dk-SQR((y-yc)^2+(z-zc)^2)
 IF z=zc THEN z=z+.0001
 wk-ATN((y-yc)/(z-zc))-((z-zc)<0)*3.1415
 z-dk*COS(wk+rk)+zc
 y-dk*SIN(wk+rk)+yc
END SUB
Maggiore:
  mag(1)=0:min(1)=320:mag(2)=0:min(2)-200:mag(3)=-200:min(3) 200
  FOR le=1 TO 3:FOR mm=1 TO nv
    IF var(le,mm)>mag(le) THEN mag(le)=var(le,mm)
    IF var(le,mm)<min(le) THEN min(le)=var(le,mm)
  NEXT:NEXT
Centro:
 c1(1)=((mag(1)+min(1))/2):c1(2)=c1(1)
 c2(1)=((mag(2)+min(2))/2):c1(3)=c2(1)
 c2(2)=((mag(3)+min(3))/2):c2(3)=c2(2)
 RETURN
Tasto:
 FOR tt=1 TO 19:IF MID$(ctr$,tt,1)=a$ THEN RETURN:ELSE NEXT
 tt=99:RETURN
fine:
 END
```

500 truccato

Ho letto dell'esistenza di un accessorio chiamato Zorro Big Blue. E' vero che comprandolo posso trasformare il mio Amiga 500 in un 2000? Potrò realmente inserirci tutte le schede create per il 2000?
(Luigi Conversin)

Sì, l'accessorio esiste, anche se non è più l'unico del genere presente sul mercato.

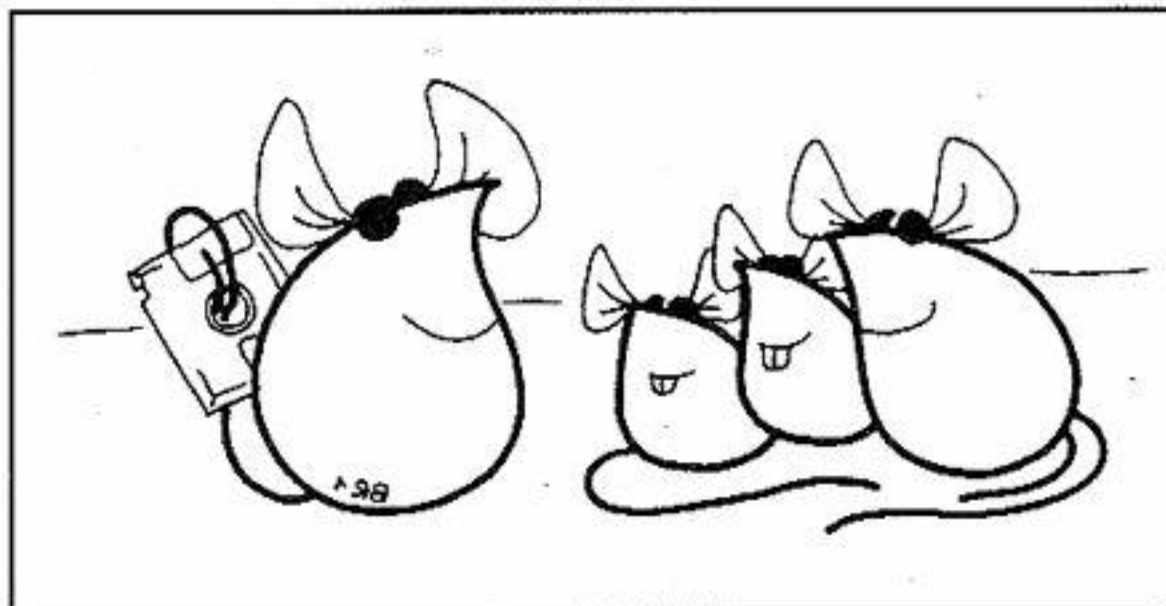
Da un po' è infatti disponibile anche un altro prodotto del genere, il **Bodega Bay**. In entrambi i casi diventa possibile sfruttare praticamente tutte le schede progettate per Amiga 2000 (har disk compresi), grazie anche ad un **alimentatore potenziato** (opzionale nel Zorro Big Blue) in grado di reggere il carico supplementare.

La scelta andrà relazionata al costo dell'operazione, senza dimenticare che Amiga 2000 ha subito un abbassamento di prezzo notevole.

Problemi con C1-Text

Pur possedendo 3 Mb di ram, con il C1-Text non riesco ad effettuare né la numerazione delle pagine, né l'inserimento di una testata, a causa di memoria insufficiente! E come potrei far apparire la data aggiornata su ogni pagina (possiedo il clock interno)?
(Alberto Canadè - Corigliano Cal.)

La memoria ram disponibile non c'entra proprio nulla, la numerazione e l'inserimento di una testata in ogni pagina si possono implementare anche con soli 512 KB!

L'equivoco, probabilmente, insorge per la segnalazione di errore che talvolta C1-Text fornisce quando si tenta di modificare i settaggi presenti nel menu Parametri/Impaginazione/Documento (raggiungibile anche con Amiga destro+D). Un esempio chiarirà meglio cosa si intende.

Supponiamo di avere appena aperto la finestra **Formato Documento**, che potrebbe presentare questa configurazione in alcuni suoi parametri:

```
Caratteri per riga: 70
Righe per pagina: 66
Righe marg. super.: 0
Righe marg. infer.: 0
```

Se tentassimo di inserire un testo nel riquadro **Testata**, questo verrebbe sicuramente rifiutato, con relativo lampeggio dello schermo a segnalare una condizione di errore. La stessa cosa avverrebbe tentando di clickare sulla posizione che dovrebbe assumere la testata (*Sinistra, Centro, Destra*) oppure su uno dei dati che riguardano il numero di pagina. Il motivo è presto detto: occorre **prima** riservare il dovuto spazio alla testata e/o al numero di pagina, per esempio settando a **2** (oppure al valore desiderato) il numero di righe sul margine superiore. In questo caso è inoltre neccesario che sia adeguatamente configurato il riquadro Superiore alla voce "Nel margine" posto in fondo alla finestra.

Stesso discorso per la paginazione: se il numero di righe sul margine inferiore è lasciato a **0**, non sarà possibile selezionare alcuna scelta ad esso relativa.

Per la cronaca, se si desidera sia una testata che la numerazione delle pagine, è necessario impostare a **2** questo parametro. Le varie possibilità di allineamento tra testata e numero di pagina può essere poi verificata sperimentalmente, se proprio si è allergici al manuale.

Quanto alla possibilità di riprodurre nell'ambito di un documento una data aggiornata, è sufficiente selezionare dal menu **Parametri / Caratteri** l'opzione **Etichette**, quindi digitare la parola **Data** nella posizione voluta. Questa etichetta, che nel testo assumerà una colorazione in rosso (a meno di cambiamenti nella palette), in fase di stampa su carta verrà sostituita dalla data impostata nel sistema.

Se la si desidera in ogni pagina, non è possibile automatizzare la procedura inserendo l'etichetta come testata, per cui si renderebbe necessario operare manualmente nel contesto del documento, che normalmente mostra chiaramente il passaggio da una pagina all'altra.

Lapsus... monetario

In una vostra recensione di giugno, affermate che Master Sound costa intorno ai 40 dollari: non si tratterà di un errore di stampa?
(Stefano Smania - Aquileia)

Aaaargh! Beccato! Non è un errore di stampa, ma proprio un lapsus del recensore (ehm...), cui l'inveterata abitudine all'unità monetaria

Speciale principianti

Facile facile...

Non riesco, da Basic, ad avere la directory del floppy "miei lavori" dal drive Df0: (ho solo quello). Come si fa?

(Diego Solventi)

La risposta è più semplice di quanto sembri. Basterà digitare nella finestra di output del Basic un comando...

`Files "miei lavori:"`

...senza dimenticare i due punti finali e le virgolette. Se il floppy non si trova già nel drive interno, apparirà un requester che ne richiede l'inserimento. Sarà sufficiente ubbidire, attendere qualche secondo, e il gioco è fatto.

Se, invece, il disco si trovava già in **Df0**, la directory verrà mostrata senza esitazione (o quasi, vista la lentezza di Amiga Basic) sullo schermo.

Un consiglio: in previsione di tempi in cui non si sarà più alle prime armi, è opportuno evitare l'uso di **spazi** nel nome di un disco (ma anche di subdirectory e files): se da Basic la cosa non crea problemi, quando ci si avventurerà tra le finestre di **Shell** si andrà incontro a difficoltà ben maggiori, anche se non insormontabili.

Un tasto come un altro

Sono passato ad Amiga dal C/64 e, quando avevo quel computer, riuscivo a gestire da basic i tasti funzione (F1, F2, eccetera) e anche i tasti cursore. Sì, ho scoperto il comando Locate di Amiga Basic, ma per gli altri?

(G.Emanuele - S.M.di Sala)

Intuendo probabilmente la soluzione al problema, la lettera del lettore prosegue lamentando l'incompletezza dell'appendice del manuale riservata alla descrizione dei codici ASCII.

La pressione dei tasti funzione, così come di quelli cursore, può infatti essere intercettata facilmente interpretando da programma, dopo una funzione di attesa **Inkey$**, il codice ASCII associato al tasto. Procedura, tra l'altro, del tutto simile a quella che si adottava sui piccoli "ex" di casa Commodore, ma con una diversa corrispondenza dei codici.

Anche se di uso poco comune proprio per l'esistenza di appositi comandi quali **Locate**, ecco comunque i codici associati ai tasti di "movimento":

```
Su ------- 28
Giù ------ 29
Destra --- 30
Sinistra - 31
```

Come tutti quelli inferiori al valore numerico **127**, e diversamente da quanto avveniva sul citato C/64, questi codici sono **realmente standardizzati**, ovvero hanno lo stesso significato anche su computer diversi da Amiga (quali i PC).

Ai tasti funzione, invece, corrispondono valori appartenenti alla fascia **non standard** dell'ASCII, costituita dai codici compresi tra **128** e **255**. In particolare, ai 10 tasti funzione sono associati i valori da 129 (**F1**) a 138 (**F10**) nell'ordine, mentre il tasto **Help** può essere sfruttato tramite il codice 139.

Da un punto di vista pratico, si consideri il banale esempio d'uso riportato nel riquadro.

Mandandolo in esecuzione, si gestiranno i tasti funzione F1, F2 ed F3, nonché il tasto Help. In questo esempio, premendo uno dei suddetti, apparirà una *stringa* che precisa il tasto premuto e il suo corrispondente codice ASCII, ma, come si può notare, il listato è schematicamente suddivisto in subroutine, ognuna contrassegnata da una etichetta riferita al tasto (F1:, F2:, eccetera): nulla vieta che, al posto dei banali Print, si programmino altri più nobili compiti eseguiti alla pressione del tasto funzione voluto, o che si crei un menu a 10 (oppure 11, con l'help) opzioni.

Le istruzioni presenti nel listatino non presentano alcuna difficoltà, per cui sarà sufficiente un'occhiata al manuale, se non le si conoscono ancora. Unico elemento (relativamente) più complesso da precisare, l'uso dell'istruzione **On ... Goto**. Questa, per funzionare correttamente, necessita in Amiga Basic che il valore cui fa riferimento sia proprio una sequenza 1, 2, 3, eccetera. Poiché i valori Ascii legati alla variabile A saranno invece 129, 130 oppure 131, si rende necessaria la variabile **A2**, che, sottraendo 128 al valore originale, conterrà appunto 1, 2, oppure 3 a seconda del tasto funzione adoperato.

Con la stessa tecnica, ma adoperando i codici ASCII prima elencati, è possibile gestire anche i tasti cursore, assegnando loro una specifica funzione.

Tutto qua.

```
CLS
PRINT"Selezione tra i"
PRINT"tasti F1, F2, F3"
PRINT"oppure Help":PRINT
PRINT
PRINT"Premine uno":PRINT

loop:
a$=INKEY$
IF a$="" THEN loop
a=ASC(a$)
IF a=139 THEN GOTO help
IF a<129 OR a>131 THEN loop
a2=a-128:CLS
ON a2 GOTO f1,f2,f3

f1:
PRINT"Hai premuto F1"
PRINT"codice ascii";a
END

f2:
PRINT"Hai premuto F2"
PRINT"codice ascii";a
END

f3:
PRINT"Hai premuto F3"
PRINT"codice ascii";a
END

help:
CLS
PRINT"Hai premuto Help"
PRINT"codice ascii";a
END
```

d'oltre oceano ha giocato un brutto scherzo: si trattava di **sterline** inglesi, non di **dollari**. Per essere pignoli (non è mai troppo tardi), il coupon di ordinazione del prodotto riportava la cifra **£ 39.95**.

Vale la pena?

Ho un problema con la mia scheda Janus At: la lentezza. Ho provato a renderla più veloce sostituendone la Cpu, acquistando un coprocessore matematico e una scheda grafica Vga (con relativo monitor), ma il miglioramento è stato minimo. Come potrei velocizzarla?
(Mario Izzo)

Una volta tanto, una lettera è pubblicata più come monito che per dare una vera risposta.

Che, per inciso, non potrebbe che essere una: acquistare un computer MS - DOS compatibile, da **affiancare** al sempre superlativo Amiga.

Passi per il monitor, teoricamente riutilizzabile (ma con i tempi che corrono andrebbe comunque preso in considerazione un **multisync**), ma cosa può giustificare una spesa che, sommata a quella della scheda, è fin troppo vicina al costo di un computer di quella categoria?

Il lettore non se ne abbia a male, possiamo capire lo sconforto, ma il suo represso lamento salverà forse qualche altra vittima predestinata...

Ogni cosa a suo tempo

Ho saputo di una nuova versione del Workbench, la 2.0. Perché non ne parlate sulla rivista?
(A. Pellegrini - Napoli)

Il cosiddetto Workbench 2.0 esiste già da tempo, ma il suo uso è strettamente legato alla **versione 2.0** del sistema operativo di Amiga, finora implementato solo sugli **Amiga 3000**. Vero è che, con molta memoria a disposizione (più di 1.5 Mb), e come già riferito in un precedente appuntamento con Postamiga, si può artificiosamente caricare in ram una copia delle Rom 2.0 e provare l'ebbrezza del nuovo DOS e del nuovo Workbench, ma la cosa non ha (al momento) riscontri pratici per la maggioranza degli utenti medi di Amiga.

Si parla, ora, di una possibile commercializzazione della 2.0 anche per i computer della fascia più bassa, che dovrebbe comportare per forza di cose la sostituzione del kickstart (che passerebbe a ben **512 Kb**). Ma finché la cosa non sarà ufficiale, e soprattutto tangibile, la politica della nostra rivista resta la stessa di sempre: prima toccare con mano, poi parlarne.

Tra l'altro, non avrebbe senso filosofeggiare su qualcosa che ancora potrebbero valutare solo i pochi (anzi, molto pochi) possessori di un 3000, che certamente non avranno speso 6 - 7 milioni per smanettare un po' con un computer...

Gli attuali utenti di 500 e 2000 dunque non temano: se il nuovo sistema divenisse realmente alla portata di tutti, diverrebbe presto argomento principe nelle nostre pagine.

Nel frattempo, il 99.99% del software esistente gira in funzione dello schiacciante parco hardware installato nel mondo, vale a dire Amiga 500 e 2000.

❑❑

Accesso riservato

Vorrei poter inserire, su richiesta, una password nella startup-sequence di un disco che non voglio mia figlia veda.
(Sergio Palumbo - Bari)

Adoperando solo i comandi del DOS, ovvero senza costruirsi un programma apposito, la cosa è facilmente fattibile, ma, come ovvio, qualunque smanettone potrebbe tranquillamente andare a leggersi la startup-sequence e vanificare il trucco.

Se invece, come sembra il suo caso, la protezione è rivolta a eventuali semplici user, in grado solo di adoperare i dischi di Amiga, ecco una possibile soluzione.

Anzitutto è necessario memorizzare da qualche parte, sotto forma di **file**, la **parola**

Brevissime

Vorrei sapere se esiste un compilatore per Amigabasic.
(Roberto Guida - Casavatore)

Sì, anche se sarebbe più corretto dire **compatibile** con Amiga basic. Il suo nome è **AC Basic Compiler**, e se ne è parlato spesso in passato anche nelle pagine della nostra rivista.

E' possibile collegare le schede di emulazione Ibm "Atonce" e "Pc Power Board" per Amiga 500 a un vecchio Amiga 1000?
(Alessandro Bonelli - Roma)

No.

E' commercializzato un compilatore per Amos?
(Marco Brivi - Milano)

Sì, in concomitanza con la versione **1.3** del relativo interprete. Se ne parlerà più dettagliatamente in altra sede.

Perché non fate come ai tempi del C/64, e scrivete uno Speciale Drive per Amiga, spiegando come sono organizzati i dati nei suoi floppy?
(S.P. - Bari)

Probabilmente prima o poi lo si farà, ma un paragone è improponibile. I drive di Amiga non hanno una memoria da disassemblare, e l'organizzazione dei dati è così complessa (tanto da descrivere che da gestire) da non renderne pratica la manipolazione, se non generica.

Molto più comodo sfruttare a fondo il software dedicato già esistente.

Il negoziante mi aveva assicurato, quando ancora non avevo Amiga, che quel modello di stampante sarebbe stato sicuramente compatibile, invece ora...
(troppi per essere citati tutti)

Prima provare, **poi** comprare!

Desidererei sapere il nome di qualche software per Amiga che consenta di accedere al Videotel Sip.
(Giovanni De Michele - Bari)

Possiamo solo citare, senza averne una esperienza diretta, due programmi circolanti nell'ambiente Pubblico Dominio: **Supertex** e **Rubi-View**. Sono facilmente reperibili in una delle tante bbs che riservano spazio ad Amiga.

d'ordine che consentirà l'accesso normale al disco. Per il nostro esempio utilizzeremo, come password, **secret**, e la nasconderemo nella directory **Devs** del disco di sistema con nome **PW**. Per farlo, il modo più semplice consiste nel digitare direttamente, da una finestra Shell, un comando...

```
Echo >devs: pw secret
```

Si dà per scontato, in questo caso, che Amiga sia stato bootato con il disco che dovrà essere protetto. In caso contrario, basterà specificare il corretto percorso. Se il disco da proteggere si trovasse in **Df1:**, allora l'istruzione prima vista diventerebbe...

```
Echo >df1:devs/pw secret
```

Alla startup-sequence, adoperando l'editor preferito, vanno poi aggiunte le righe che seguono:

```
;makedir ram:env
;assign env: ram:env
copy devs:pw env:
echo "Parola ordine?*N"
setenv >nil: name=x ?
if $x not eq $pw
 echo "*NNon puoi!*N"
 quit
endif
echo "*NBenvenuto!*N"
;qui continua la normale
;startup-sequence
```

Le prime due righe di istruzioni, si noterà, sono impostate come commento dal punto e virgola (;) iniziale, quindi non attive. Il motivo è presto detto: il batch file, per funzionare correttamente, necessita che sia già stato assegnato il device **env:**, che gestisce le variabili di sistema. In altre parole, se nella startup-sequence è già presente una istruzione **Assign Env:**, allora le due righe non si rendono necessarie, e il batch file andrà inserito **dopo** quella istruzione. In caso contrario, basterà togliere il simbolo di punto e virgola alle prime due righe, e inserire il batch in qualunque punto della startup-sequence (preferibilmente all'inizio). Il funzionamento è (quasi) banale: dopo lo start del disco, il nostro batch copia il file **pw** (contenente la giusta password) nel device **env:**, facendolo in pratica diventare una variabile dello stesso nome. A questo punto, chiede l'immissione della password da tastiera (che va fatta seguire dal Return), e la assegna alla variabile **$x**. Confronta poi **$x** e **$pw**, e se non sono uguali arresta la procedura di start. Se, invece, si è digitato **secret**, viene eseguito il resto della startup - sequence, ovvero quella originale del dischetto.

Inutile aggiungere che, al posto di Secret, può essere adoperata qualunque altra parola, badando che, se si tratta di una frase con degli spazi, va racchiusa tra virgolette quando la si memorizza con Echo (per esempio: `Echo >devs:pw "ti vedo solo io"`).

Come ovvio, la directory **C** del disco dovrà contenere tutti i comandi adoperati dalla sezione di startup-sequence sopra descritta.

Il batch, a parte il suo uso, costituisce una buona base sulla quale esercitarsi nell'uso del DOS. Un miglioramento, per esempio, potrebbe essere la ripetizione della richiesta di password un numero prestabilito di volte, in modo da evitare che, se si commette un errore di digitazione la prima volta, sia necessario resettare il computer per far ripartire il disco.

Sorpresa svelata

Che cosa significano quei riquadri neri sul n. 85, che promettevano qualcosa di nuovo a partire da settembre?
(da lettere e telefonate)

Beh, ormai ci siamo, si può anche cominciare a parlarne. Si tratta in pratica di un annuncio funebre, ma del tipo *"è morto il re, lunga vita al re"*.

Chi defunge, in un certo senso, è la serie di floppy finora disponibili su richiesta dei lettori, che lascia spazio ad un nuovo evento: la nascita di **Computer Club Disco**, per gli amici CCD (si torna a vecchie assonanze, vero?). CCD sarà una pubblicazione su floppy, direttamente acquistabile in edicola, da considerare **gemellata** alla nostra rivista. Nel senso che l'integrazione con Computer Club sarà totale, così come lo scambio di favori tra le due testate.

Il floppy, che aspira soprattutto a essere **diverso da quanto già presente in edicola**, proporrà anche software di pubblico dominio degno di finire tra le mani del nostro lettore, ma con una caratteristica: sarà sempre integrato da **facilitazioni d'uso per principianti e non.** Il che significa, per esempio, che se un certo programma può risultare difficile da installare nel sistema, penseremo noi a elaborare una procedura automatica di installazione, e inserirla nel disco.

Questi interventi, inoltre, non saranno **mai** solo fine a se stessi, ma verranno approfonditi a scopo didattico nelle pagine della solita rivista (cioè questa), in modo che ognuno possa personalizzare a proprio piacimento l'uso del software.

Per di più, e questa è forse la cosa più importante, si instaurerà una nuova abitudine su queste pagine.

Talvolta, e proprio in questo numero di CC se ne possono constatare due esempi, i listati pubblicati raggiungono dimensioni che sicuramente non invogliano il lettore a lunghe e noiose sessioni di copia, tra l'altro con probabilità piuttosto alte di incorrere in banali errori di digitazione.

Questi listati, d'ora in poi, finiranno su **Computer Club Disco**, mentre gli articoli espicativi continueranno ad apparire sulla rivista cartacea.

Il che significa, anche, un allargamento delle possibilità di collaborazione dei lettori. Finora, infatti, molti lavori (o risposte alle varie sfide) erano esclusi dalla divulgazione proprio perché improponibili a causa della loro lunghezza. **In altre parole, ora potrete scatenarvi!**

Sulla rivista, naturalmente, continuerete a trovare le piccole routine o gli esempi esplicativi di particolari procedure, che comunque saranno presenti **anche** sul floppy CCD, per la gioia dei più pigri.

A garantire il massimo dell'interazione, i vostri dubbi, problemi, o anche suggerimenti, saranno come sempre ospitati nelle rubriche di posta di Computer Club, anche se riguarderanno CCD.

Computer Club Disco non sarà solo questo, altre novità sono in cantiere, ma... è già argomento da trattare in altra sede, nonché direttamente su CCD.

Provare per credere e... arrisentirci su floppy!

Miglioriamo la grafica in Pascal

< di Filippo Bosi >

< Esaminare C. C. n. 85 e 86 > < Si lancia anche una sfida ai lettori! >

Continuiamo a scrivere la nostra Unit di comandi grafici tipo **BGI** (Borland Graphic Interface, quella originale **Borland**, per intenderci) aggiungendo **nuovi comandi**, ma anche migliorando quelli già costruiti, plasmando il nostro insieme di routine in modo da emulare sempre meglio TP. Giunti alla seconda "edizione" della scrittura di **TPGrafica** ci troviamo di fronte ad un problema: le aggiunte al codice sorgente **non** vanno fatte in fondo al listato, come se avessimo scritto il mese scorso il primo capitolo di un romanzo e ci accingessimo a scrivere **il** secondo...

Aggiungere una routine significa aggiungere variabili ed anche aggiungere una sua dichiarazione nella sezione **interface** (nel caso debba essere utilizzata da programmi che richiamano la unit).

Per evitare guai, o meglio per cercare di *limitarli* al massimo, errori di battitura permettendo, l'optimum sarebbe quello di pubblicare l'intero sorgente di TPGrafica tutti i mesi sulle pagine della rivista. In questo modo, però, occuperemmo una parte consistente delle pagine, essendo costretti, ogni mese, a pubblicare nuovamente una buona parte di codice.

D'altra parte, riportare le sole modifiche nel codice sorgente, non avendo il Pascal (e quasi tutti i moderni linguaggi) numeri di linea a mò del Basic, porterebbe a qualcosa che possiamo definire come caos totale nelle menti dei lettori e nei risultati delle compilazioni.

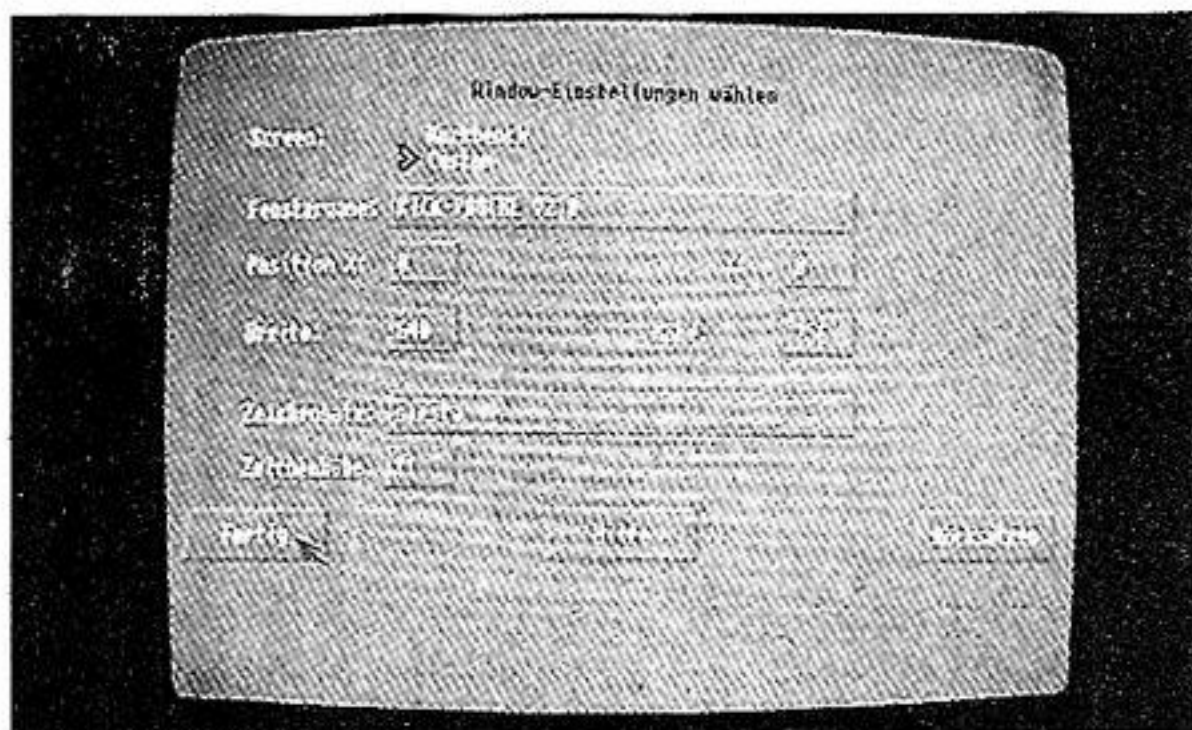
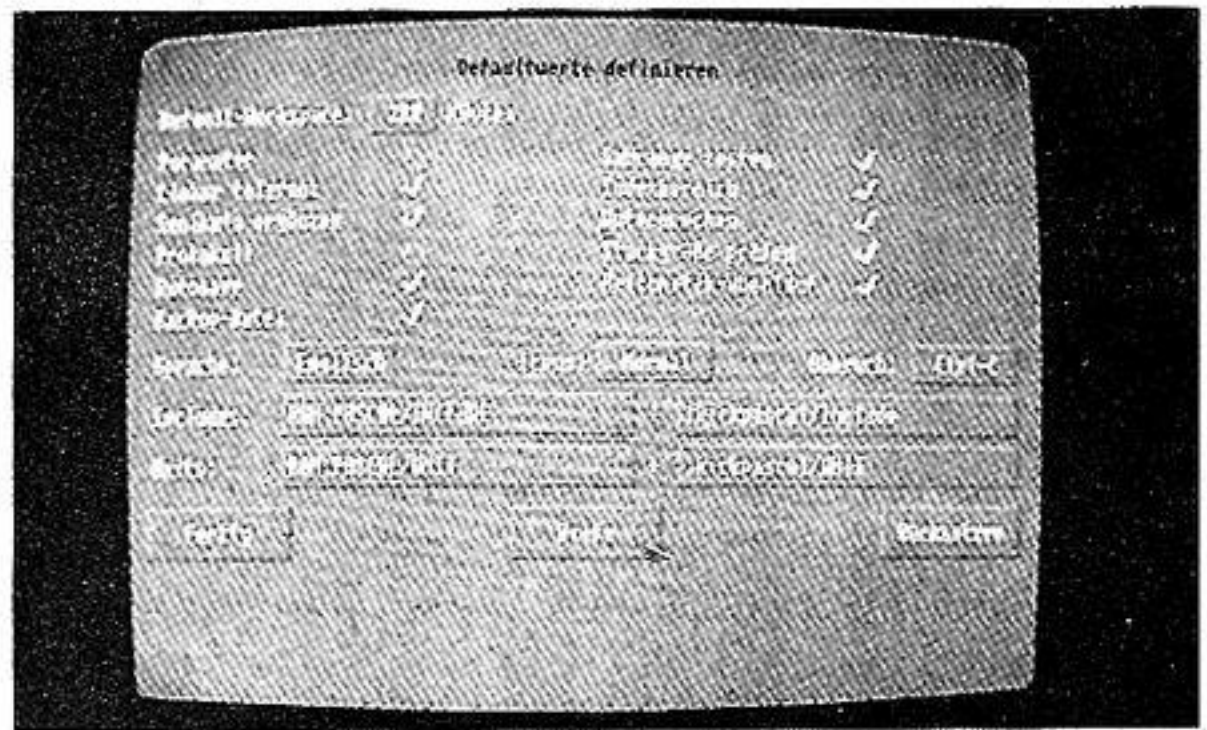
Ogni tanto, quindi, verranno pubblicati **aggiornamenti** alla unit, strutturati in questo modo: verrà pubblicata interamente ogni sezione del programma aggiunta o anche solo aggiornata, con al suo interno evidenziate (per quanto possibile) le parti che hanno subito modifiche.

Ad esempio, nel listato di questo mese è riportata *interamente* la routine **Init-**

K. Pascal, dove trovarlo

Numerosi lettori ci hanno scritto chiedendoci come procurarsi una copia del compilatore Kick Pascal. Un gruppo di Amighi, nell'esprimere la propria gioia nel veder pubblicati articoli su KP, affermano tra l'altro di averlo trovato presso un Computer Club di **Imola** (Bologna) ed invitano i neofiti della zona a procurarselo al più presto. Riteniamo che, non essendo Imola una grossa metropoli, il compilatore sia facilmente reperibile anche presso qualsiasi buon negozio di informatica, oppure per corrispondenza, da una delle tante ditte specializzate. A mano a mano, comunque, che avremo maggiori informazioni in merito, ne daremo ampia diffusione.

Dedicato a chi inizia

Sui n. **85** e **86** di **Computer Club** sono stati affrontati alcuni problemi relativi al **Kick Pascal**, potente compilatore per **Amiga**, in grado di "girare" anche sulla versione base del popolare computer (500 KRam, un solo drive).
In particolare, si è cercato, e si cercherà in futuro, di rendere ***compatibili*** i programmi sorgenti scritti in ambiente MS - DOS con quelli per **Amiga** e viceversa.
Lo scopo, inutile dirlo, è quello di consentire l'utilizzo del potente compilatore da parte degli studenti che, abituati a scuola ad usare sistemi MS - DOS, possano esercitarsi, a casa propria, usando l'Amiga.

Configurazione dei parametri video

Configurazione dei parametri di default

PascalPrefs, ultimo atto

Eccoci giunti alla seconda ed ultima tappa del nostro viaggio all'interno di PascalPrefs, il programma di **configurazione** di KickPascal.

Dopo avere affrontato lo schermo **Window Einstellungen** (punto **6** dello schema di pagina **74** apparso sul n. **86** di C.C.), passiamo alla scelta dei parametri di default dell'ambiente integrato (opzione **5** dal menu principale), alla configurazione del file requester (opzione **4**) ed infine alla configurazione delle sequenze di tasti dei comandi dell'editor (opzione **3** dal menu principale).

Defaultwerte definieren

Sempre nell'ordine, dall'alto verso il basso, appaiono le seguenti voci:

⇨ **Default workspace:**

Di fianco si deve specificare la quantità di memoria resa disponibile all'ambiente integrato per contenere il listato sorgente ed il codice oggetto ottenuto dalla compilazione.

Ricordiamo che (purtroppo) questo spazio viene riservato in **Chip Memory**, quindi attenzione a non esagerare, specialmente se non si possiede il nuovo chip **Agnus** che rende disponibile 1Mb di memoria Chip, pena l'impossibilità di far girare altri programmi in multitasking, anche se disponiamo di svariati "mega" di memoria libera.

Bisogna infatti sempre tener presente che, esaurita la memoria Chip, non si possono più aprire nuovi schermi e nuove finestre, anche se si hanno a disposizione bytes liberi in memoria Fast.

Sotto troviamo una serie di "bottoni" mediante i quali impostare, come default all'inizio di ogni sessione di lavoro, molte opzioni che possono essere scelte anche tramite i menu all'interno dell'ambiente integrato KP.

Molto importanti sono:

Autosave: salvataggio automatico del sorgente (se modificato) prima di ogni partenza del programma compilato. Opzione molto utile, in quanto può capitare che il nostro programma non sia estremamente corretto e faccia andare la macchina in Guru, non permettendo magari più di salvare il risultato di ore di duro lavoro.

Stackgrosse Prufen: test di un eventuale Stack Overflow all'entrata di ogni subroutine.

Arithmetik Uberlauf: test di Overflow aritmetico.

Backup Datei: scrittura automatica di una copia di backup della versione precedente di ogni sorgente che si salva.

Seguono altri Gadget:

⇨ **Sprache**:

Serve a scegliere il linguaggio utilizzato dall'ambiente integrato; si consiglia di scegliere tassativamente "Englisch", cioè inglese, a meno che non siate esperti conoscitori della lingua tedesca(!).

⇨ **Icons**:

Ein Ogni sorgente e ogni eseguibile che salviamo avrà automaticamente una sua icona.

Aus Viene salvato esclusivamente il file, quindi da eseguire da CLI.

Normal L'opzione di salvataggio con icona viene attivata se si fa partire l'ambiente integrato da Workbench (ovvero con l'icona KP), invece che da CLI con il comando KP.

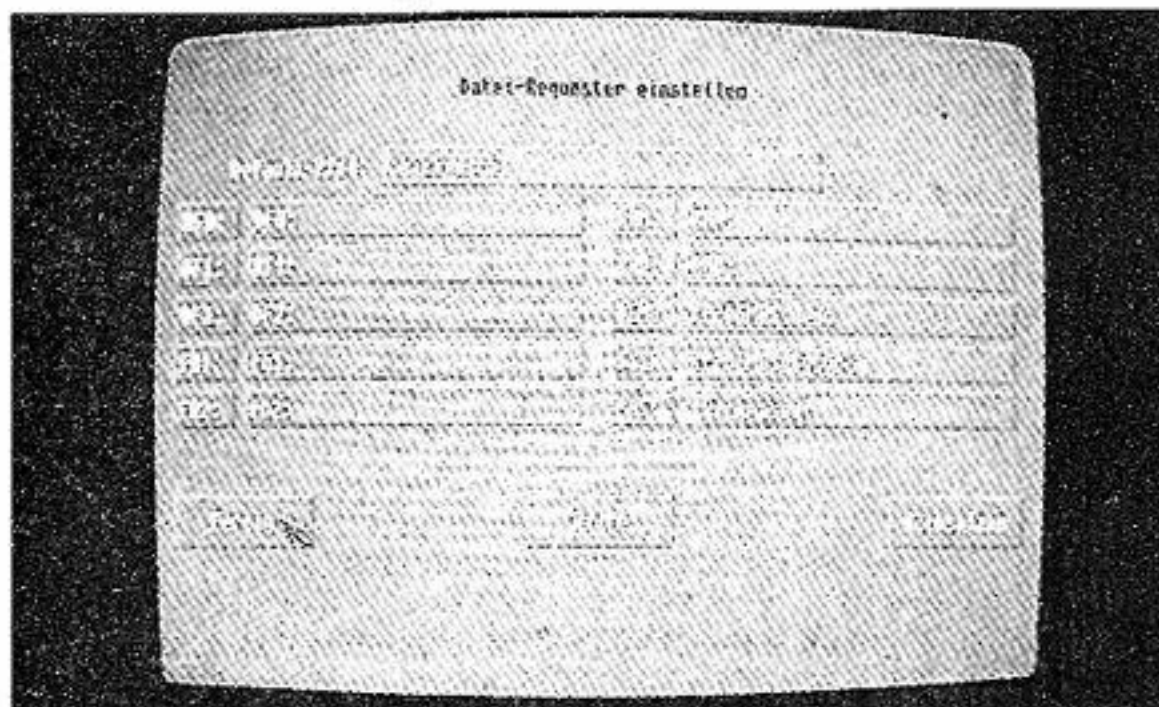
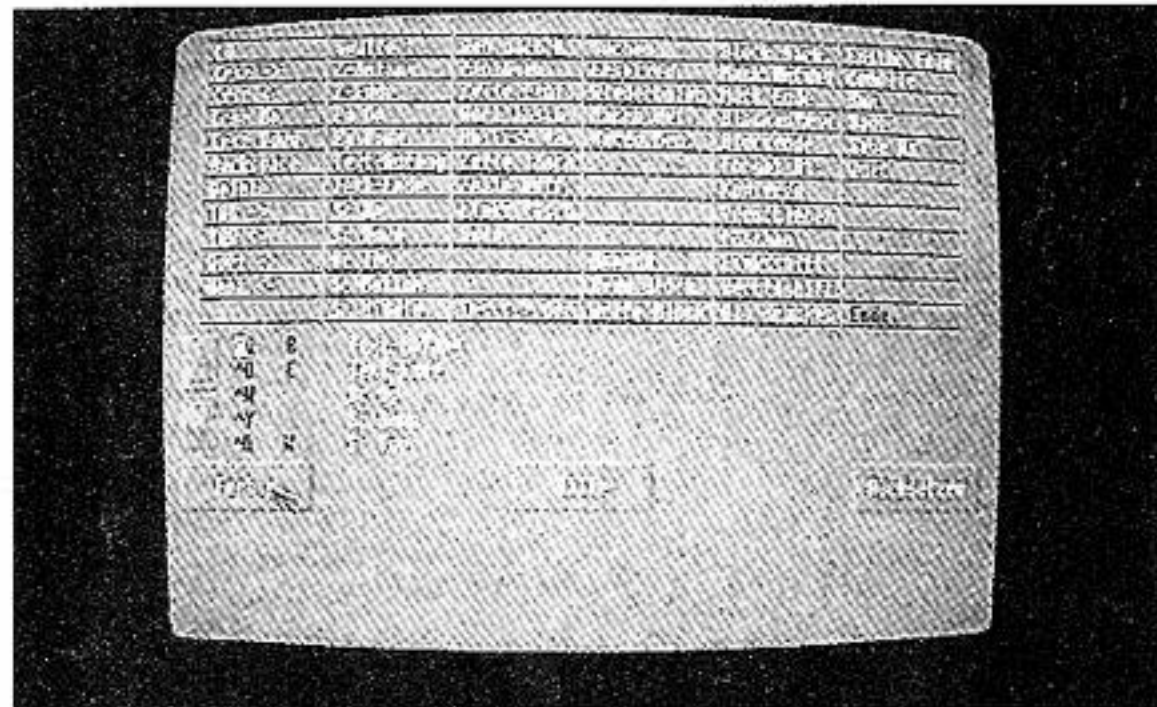
⇨ **Abbruch**:

Si sceglie quale sequenza da tastiera valga come break per i programmi KP: **F10** oppure **Control + C**.

⇨ **Includes**:

Alla destra di questa scritta appaiono due gadget, separati da un segno più (**+**): sono i percorsi di ricerca del compilatore KP per i files include, cioè i files che realizzano l'interfacciamento dei nostri programmi sorgente con le librerie di sistema di Amiga.

Il compilatore cerca i file include *prima* nella directory indicata dal gadget di sinistra, *poi* in quella del gadget di destra.

Si può notare come per default nel gadget di sinistra siano contenuti percorsi in RAM:, questo perché se il compilatore trova i file include solo nella directory di destra, esso automaticamente copia i file cercati nella directory di sinistra, cioè nel ram disk (memoria permettendo), rendendo di fatto molto più veloci le compilazioni successive dello stesso program-

Configurazione del file requester

Configurazione dei tasti dell'Editor

ma (o di programmi che utilizzino gli stessi file include appena copiati).

Bisogna quindi specificare un percorso su Ram Disk nel gadget di sinistra (esempio: **ram:pascal/include**) e il percorso su disco dove effettivamente abbiamo i file (esempio: **kickpascal1:pascal/include**, oppure **df0:pascal/include**, a seconda della nostra configurazione di sistema).

⇨ **Units:**

Come per i gadget alla destra di includes dobbiamo indicare di seguito due percorsi di ricerca, questa volta però per le Unit (esempio **ram:pascal/unit + kickpascal1:pascal/unit**).

File requester einstellen

E' la configurazione del file requester. Esaminiamo in dettaglio tutti i parametri.

⇨ **Default pfad:**

A destra di questo gadget si deve inserire il percorso per la directory di default del file requester.

Sotto troviamo **20 gadget**, accoppiati a due a due, uno corto ed uno più lungo. Indicano, rispettivamente, il contenuto dei bottoni del file requester e il percorso corrispondente: ad esempio, se scrivo in un gadget piccolo PAS e nel grande, a fianco...

`dh0:pascal/sources`

...apparirà ogni volta, nel requester del KP, un bottone PAS, premendo il quale verrà visualizzata la directory *pascal/sources* del disco **dh0:** (generalmente un hard disk).

Editor-tasten belegen

Si tratta, più banalmente, della configurazione dei comandi dell'editor. Scelta questa opzione da menu, ci troviamo in uno schermo con **72 bottoni** (rettangolari, ognuno con all'interno un messaggio corrispondente ad una operazione che è possibile effettuare nell'editor, premendo una opportuna sequenza di tasti).

Sotto, a sinistra, appaiono **due frecce**, una rivolta verso l'alto, l'altra verso il basso: servono a fare scorrere una lista di sequenze di tasti visualizzate subito a destra (in nero).

Ad ognuna delle sequenze corrisponde una delle azioni riportate all'interno dei 72 bottoni in cima allo schermo, visualizzata a destra della sequenza dei tasti (in colore bianco).

Aggiungere una nuova sequenza è semplice come aggiungere una voce di glossario in un moderno word processor.

In altre parole: premere il tasto **"i"**, immediatamente verrà creato spazio per inserire la sequenza voluta, che potrà esser costituita al massimo da **3** tasti.

Battuti i 3 tasti (che non appartengano, ovviamente, ad una sequenza prima definita) dobbiamo scegliere una certa azione, che vogliamo venga eseguita con la pressione di detti tasti, premendo uno dei bottoni in alto.

Quando torneremo al Kick Pascal, avremo a disposizione un comando in più (o, meglio, un modo in più per eseguire un comando).

Se invece desideriamo eliminare una sequenza, allora basta premere **d** oppure il tasto **Del**, dopo averla selezionata con il mouse e le frecce.

Le sequenze di comando possono contenere, oltre alla maggior parte delle sequenze Ctrl + lettera (indicate come **^lettera**) anche tasti funzione ed il tasto **ESC**. Ad esempio, una sequenza potrebbe essere: **F1, Control + K, Control + B.**

Fanno eccezione alcune sequenze di tasti: caratteri non validi all'interno di una sequenza sono infatti **Control + H, Control + I** e **Control + M**, per il semplice fatto che corrispondono, rispettivamente, ai tasti **Backspace, Tab** e **Return.**

I messaggi posti all'interno dei bottoni sono in tedesco (a parte alcune eccezioni tipo Run, Save, ecc.).

Per una loro traduzione rimandiamo ad una tabella che pubblicheremo prossimamente.

In ogni caso ci si può già arrangiare guardando la tabella dei comandi dell'editor KP pubblicata nel numero 86 (Luglio / Agosto), confrontando le sequenze di tasti ivi riportate con quelle visualizzate da Pascalprefs, a meno che non vi siate già sbizzarriti a modificarle tutte!

Utenti di Pascal, unitevi!

Filippo Bosi è un valido collaboratore la cui presenza, sulle pagine della nostra rivista, inizia a diventare assidua, anzi costante.

La sua specialità è il linguaggio Pascal, sia nell'implementazione **Turbo Pascal Borland** sia in quella **Kick Pascal**. Bosi, infatti, appartiene alla fortunata(!) percentuale (vedi, in altra parte di questo stesso fascicolo: *I risultati dell'inchiesta*) che possiede entrambi i sistemi, **Amiga** ed MS - DOS.

Come ogni appassionato "puro", tra l'altro, non desidera essere l'unica voce a parlare nello spazio dedicato al Pascal e si dichiara completamente aperto a suggerimenti, proposte di modifiche al codice e tutto quanto può risultare costruttivo per il nostro viaggio all'interno del compilatore.

Nell'articolo di questo mese ha focalizzato l'attenzione sul codice della routine **Arc**, ma qualsiasi altra parte del codice di **TPGrafica** può essere oggetto di approfondimento e miglioramento, grazie al vostro aiuto.

Inviate quindi lettere, listati in T. Pascal oppure K. Pascal (ma, in questo caso, solo su **disco**) presso l'indirizzo della nostra Redazione...

Systems Editoriale
COMPUTER CLUB
All'attenzione di Filippo Bosi
Via Mosè, 22
cap 20090 OPERA (Mi)

Sarà nostra cura consegnargli, in tempi ristretti, tutta la corrispondenza che vorrete inviare.

Graph (vedi C.C. n. 86 pag. 78) ma modificata in modo da comunicare con una nuova routine (**GetMaxColor**).

Dovrete quindi confrontare InitGraph del n. 86 ed effettuare le dovute modifiche.

Ciò vale anche per le altre sezioni del programma, oltre a quelle delle routine. Ad esempio, nel listato di questo mese troverete tutte le dichiarazioni, da inserire nella sezione **Interface**, delle nuove routine e delle nuove variabili.

A proposito di routine...

Nel listato di questo mese si trovano routine per la gestione del **cursore grafico**, ovvero un punto *immaginario* sul nostro schermo, punto che possiamo muovere a piacere con la routine **MoveTo** oppure **MoveRel**.

Con il cursore possiamo anche tracciare linee (**LineTo** e **LineRel**) e, se per caso, ad un certo punto del programma abbiamo bisogno di sapere dove è andato a finire il cursore a forza di spostamenti, niente paura: sono a nostra disposizione **GetX** e **GetY** pronte a rispondere ai nostri dubbi.

Ciò che preme sottolineare, a proposito delle routine di cui sopra, è che il Cursore Grafico può essere considerato una forma di ***Data Abstraction***, termine informatico (ovviamente in inglese) che indica la creazione di una entità logica (in questo caso, il Cursore) accessibile solamente mediante procedure (**MoveTo**, **LineTo**, **GetX** ed altre) indipendenti dalla implementazione a livello più basso.

Questo tipo di programmazione può risultare vantaggioso in molti casi: ad esempio potremmo decidere, fra qualche mese, di riscrivere l'intero insieme di routine che gestiscono il cursore grafico, lasciando inalterata però la sezione **interface** (interfaccia con il programma che richiama la unit), vale a dire i tipi dei parametri passati. In questo caso tutti i programmi precedentemente creati, e che comandavano il cursore grafico mediante procedure standard, non accedendo alla sua effettiva implementazione (cioè alle variabili **CPx** e **CPy**) funzioneranno ancora e beneficieranno delle modifiche apportate al codice di gestione del Cursore.

Uno dei grossi punti di forza delle Unit del Turbo e del Kick Pascal è quello di facilitare di molto questo tipo di programmazione che, tra le altre cose, permette di scrivere facilmente codice indipendente dalla macchina, anche se a spese di una minore efficienza.

Più di una sfida

Ma torniamo al nostro codice di **Tpgrafica** per lanciare una sfida fra i lettori.

Osservando la routine **Arc** al lavoro, nel programma **Arcs.P**, vediamo come sia lenta, soprattutto in confronto al tracciamento di circonferenze eseguito dalle librerie di sistema Amiga. In effetti Arc, così come è stata scritta, non è un capolavoro di efficienza. Sarebbe bello vedere la stessa routine, ma costruita da voi in modo da ottenere velocità di tracciamento comparabili a quelle di **Circle**.

Come premio ai lettori più bravi, fama e gloria su questo spazio dedicato al Kick Pascal e la propria routine inserita all'interno di TPGrafica!

```
* Sintassi e descrizione delle procedure *
* e funzioni rese disponibili dalla Unit *
* TPGRAFICA (v1.0)                       *
* per COMPUTER CLUB    SETTEMBRE '91
* NUOVE FUNZIONI e PROCEDURE             *

------------------------------------------
N.B: Procedura (parametri)(event. risposta)
------------------------------------------
Procedure Arc(x,y:integer; angoloPart,
              angoloFine,r:word)

Traccia un arco di cerchio a partire
dall'angolo angoloPart fino all'angolo
angoloFine, di raggio r, centrato nel
punto di coordinate (x,y).
------------------------------------------
Procedure Circle(x,y:integer; r:word)

Traccia una circonferenza alle coordinate
(x,y) di raggio r.
------------------------------------------
Function GetMaxColor:word;

Ritorna il numero piu' alto che puo' esse-
re passato come colore nel modo grafico
corrente.
------------------------------------------
Function GetPixel(x,y:integer):word

Ritorna il valore del pixel di coordinate
(x,y).
------------------------------------------
Function GetX:integer;
Ritorna la coordinata X del cursore
grafico.
------------------------------------------
Function GetY:integer;

Ritorna la coordinata Y del cursore
grafico.
------------------------------------------
function GraphErrorMsg(ErrorCode:integer)
                                   :string

Ritorna una stringa di messaggio di errore
per il codice di errore grafico contenuto
in ErrorCode. (Codice ritornato da
GraphResult).
------------------------------------------
Procedure LineRel(Dx,Dy:integer)

Traccia una linea dalla posizione corrente
del cursore grafico fino al punto distante
Dx pixel sull'asse x e Dy pixel sull'asse
y.
------------------------------------------
Procedure LineTo(X,Y:integer)

Traccia una linea dalla posizione corrente
del cursore grafico fino al punto di
coordinate assolute (x,y).
------------------------------------------
Procedure MoveRel(Dx,Dy:integer);

Sposta il cursore grafico dalla posizione
corrente di Dx pixel sull'asse x e Dy
pixel sull'asse y.
------------------------------------------
Procedure MoveTo(x,y:integer);

Muove il cursore grafico fino alle
coordinate (x,y).
------------------------------------------
Procedure PutPixel(x,y:integer;
                   pixel:word)

Scrive un pixel alle coordinate (x,y).
------------------------------------------
Procedure Rectangle(x1,y1,x2,y2:integer)

Disegna un rettangolo utilizzando lo stile
di linea e il colore correnti.
------------------------------------------
Procedure SetWriteMode(WriteMode:integer)

Seleziona il modo corrente di scrittura:

WriteMode puo' essere:

- NormalPut: modo di scrittura normale
- CopyPut:   modo di scrittura normale
- XORPut: xor con i pixel gia' tracciati
- OrPut          (*)
- AndPut         (*)
- NotPut: not con i pixel gia' tracciati

(*) NOTA: il comportamento dei modi di
          scrittura AndPut e OrPut non e'
          identico a quello del Turbo
          Pascal.
          Per gli altri modi, invece, le
          funzionalita' sono identiche.
------------------------------------------
N.B. in testa alla sintassi di ogni
     funzione appaiono i seguenti
     identificativi:
------------------------------------------
     (TURBO) --> la procedura/funzione
                 svolge un compito analogo
                 alla omologa funzione
                 presente in TURBO PASCAL
   + (PARAMETRI) --> compatibilita'
                   anche a livello di
                   parametri con TP.
   + (RISPOSTA)  --> compatibilita' a
                   livello di ritorno
                   valore (nel caso di
                   funzioni).
------------------------------------------
 (AMIGA)          > funzione specifica della
                  implementazione su Amiga.
```

Descrizione delle nuove routine.

```
{  TPGRAFICA.p                            }
{  Unit di supporto per la grafica tipo   }
{  Turbo Pascal v.4.0/5.0 Ms-Dos          }
{  MODIFICHE PER IL NUMERO DEL MESE DI    }
{  SETTEMBRE di COMPUTER CLUB             }
{  by Filippo Bosi (v1.0) 30-Giugno-1991  }
{ * MODIFICHE ALL'INTERFACCIAMENTO CON  * }
{ * LE LIBRERIE DI SISTEMA AMIGA:       * }
{ * cancellare le due direttive         * }
{ * ($include graphics)                 * }
{ * ($include intuition)                * }
{ * inserire la seguente istruzione,    * }
{ * all'interno della sezione           * }
{ * Interface                           * }
{ *  Uses Graphics,Intuition;           * }
{ ************************************** }
{ *  MODIFICHE ALLA SEZIONE INTERFACE   * }
{ aggiunta di nuove costanti di errore    }
{ grafico ritornate da GraphResult        }
{ (sostituire alle vecchie)               }
{ CREATE: Luglio 91                       }
{ MODIF.: Settembre 91                    }
const
 GrOk                 = 0;   { Tutto OK }
 GrError              = -11; ( Errore  }
                             { Generico }
 GrNoInitGraph        = -1;
 GrNotDetected        = -2;
 GrFileNotFound       = -3;
 GrInvalidDriver      = -4;
 GrNoLoadMem          = -5;
 GrNoScanMem          = -6;
 GrNoFloodMem         = -7;
 GrFontNotFound       = -8;
 GrNoFontMem          = -9;
 GrInvalidMode        = -10;
 GrIOError            = -12;
 GrInvalidFont        = -13;
 GrInvalidFontNum     = -14;

{ ----- Costanti per SetWriteMode ------ }
{ - si noti come NormalPut e CopyPut     }
{   siano sinonimi.                      }
{ CREATE: Settembre 1991                 }

 NormalPut     = 0;
 CopyPut       = 0;
 XORPut        = 1;
 OrPut         = 2;
 AndPut        = 3;
 NotPut        = 4;
{ ************************************** }
{ *      nuove procedure e funzioni    * }
{ - aggiungere dopo le altre all'interno }
{   della sezione interface              }
{ CREATE: Settembre 91                   }

Procedure Arc(x,y:integer; angoloPart,
              angoloFine,r:word);
Procedure Circle(x,y:integer; r:word);
Function  GetMaxColor:word;
Function  GetPixel(x,y:integer):word;
Function  GetX:integer;
Function  GetY:integer;
Function  GraphErrorMsg(ErrorCode:integer)
                               :string;
Procedure LineRel(Dx,Dy:integer);
Procedure LineTo(x,y:integer);
Procedure MoveRel(Dx,Dy:integer);
Procedure MoveTo(x,y:integer);
Procedure PutPixel(x,y:integer;
                   pixel:word);
Procedure Rectangle(x1,y1,x2,y2:integer);

Procedure SetWriteMode(WriteMode:integer);

{ ************************************** }
{ MODIFICHE ALLA SEZIONE IMPLEMENTATION  }
{ *   aggiunte alla sezione variabili  * }
{ * coordinate x e y cursore grafico * }
{ * Create: SETTEMBRE 91             * }
{ ---------------------------------- }
CPx,CPy:integer;
{ * numero di piani di bit utilizzati * }
{ * dallo schermo grafico aperto    * }
{ * attualmente: e' settato dalla   * }
{ * routine InitGraph e viene       * }
{ * utilizzato dalla routine        * }
{ * GetMaxColor.                    * }
{ ---------------------------------- }
{ * Create: SETTEMBRE 91            * }
{ ---------------------------------- }
NumPiani:word;

{ ************************************** }
{ * MODIFICHE/AGGIUNTE alle procedure  * }
Procedure InitGraph;
{ * apre lo schermo grafico         * }
{ * Creata: LUGLIO 91               * }
{ * Modif.: SETTEMBRE 91            * }

var
 risx,risy : integer;
 maxplanes : integer;
 scrmode   : word;
 { * dimensioni finestra * }
 fin_largh,fin_alt : integer;
const
 { * altezza in pixel del titolo * }
 { * dello schermo grafico       * }
 SCREENBAR = 10;

begin
scrmode:=GENLOCK_VIDEO;

if (modo and GR_LORES)<>0 then
 begin
 maxplanes:=6;
 risx:=320;

  { * se si richiedono 6 bitplanes * }
  { * in bassa risoluzione, allora * }
  { * vai in modo 64 colori (EHB)  * }
 if npiani=6 then
  scrmode:=scrmode+EXTRA_HALFBRITE;

 end

   else begin
 maxplanes:=4;
 risx:=640;
 scrmode:=scrmode+HIRES;
        end;

{ * controlla se si  scelto il modo * }
{ * interlacciato                   * }
if (modo and GR_LACE)<>0
      then begin
           risy:=512;
           scrmode:=scrmode+LACE;
           end
      else risy:=256;

{ * controlla che il numero dei piani * }
{ * di bit non sia superiore al max.  * }
{ * consentito                        * }
if npiani>maxplanes then
```

```
begin
 GrResult:=GrError;
 exit;
 end;

Scr:=Open_Screen(0,0,risx,risy,
                 npiani,3,1,
                 scrmode,
                 ^TitoloSchermo);

{ * controlla che lo schermo sia * }
{ * stato aperto correttamente   * }
if Scr=NIL then
 begin
 GrResult:=GrError;
 exit;
 end;

fin_largh:=risx;
fin_alt:=risy-SCREENBAR;

Win:=Open_Window(0,SCREENBAR,fin_largh,
                 fin_alt,1,0,BORDERLESS+
                 GIMMEZEROZERO+
                 ACTIVEWINDOW,
                 Nil,Scr,fin_largh,
                 fin_alt,fin_largh,
                 fin_alt);

{ * controlla che la finestra sia * }
{ * stata aperta correttamente    * }
if Win=NIL then
 begin
 GrResult:=GrError;
 exit;
 end;

rp:=Win^.Rport;

{ * sistema i contenuti delle          * }
{ * variabili globali per dimensioni * }
{ * schermo grafico                    * }
MaxX:=fin_largh-1;
MaxY:=fin_alt-1;

{ * sistema il colore della penna * }
{ * corrente                      * }
SetColor(1);

{ * sistema la variabile globale * }
{ * NumPiani = n.piani di bit    * }
{ * dello schermo grafico aperto * }
{ * AGGIUNTA: SETTEMBRE 91       * }
NumPiani:=npiani;

end;

Procedure SetWriteMode;
{ -- Seleziona il modo corrente di    -- }
{ -- scrittura su pagina grafica      -- }
{ - NOTA: attualmente i modi di        - }
{   scrittura che emulano AndPut e       }
{   e OrPut non sono identici a          }
{   quelli del TURBO PASCAL              }
{ Creata: SETTEMBRE 1991                 }

begin
 Case WriteMode of
   NormalPut,CopyPut: SetDrMd(rp,JAM1);
   XORPut     : SetDrMd(rp,COMPLEMENT);
   NotPut     : SetDrMd(rp,INVERSVID);
   AndPut,OrPut: SetDrMd(rp,JAM2);
   end;
end;

Procedure Arc;
{ - Traccia un arco di circonferenza  - }
{ - NOTA: la routine potrebbe essere  - }
{         ottimizzata in molti modi   - }
{         ma cio' esula dallo scopo   - }
{         di questi articoli.         - }
{ Creata: SETTEMBRE 1991                }
var i:word;
    rd,temp:real;

begin
rd:=angoloPart/180*pi;
Move(rp,x+trunc(r*cos(rd)),
       y+trunc(r*sin(rd)));

rd:=180/pi;

for i:=angoloPart+1 to angoloFine do
    begin
    temp:=i/rd;
    Draw(rp,x+trunc(r*cos(temp)),
           y+trunc(r*sin(temp)));
    end;
end;

Procedure Circle;
{ - Disegna una circonferenza          - }
{ Creata: SETTEMBRE 1991                 }

begin
{ - nelle librerie grafiche di Amiga  - }
{ - non esiste una routine che tracci - }
{ - una circonferenza, allora facciamo - }
{ - disegnare una ellisse con i due   - }
{ - raggi uguali...                   - }
DrawEllipse(rp,x,y,r,r);
end;

Function  GetMaxColor;
{ * ritorna il num.massimo di colore  * }
{ * possibile per lo schermo corrente * }
{ Creata: SETTEMBRE 1991                }
begin
GetMaxColor:=
     Round(Exp(NumPiani*Ln(2)))-1;
{ - N.B. Exp(2*Log(NumPiani)) equivale }
{        ad elevare 2 alla NumPiani    }
end;

Function  GetPixel;
{ -- ritorna il contenuto del pixel  -- }
{ -- alle coordinate x,y             -- }
{ Creata: SETTEMBRE 1991                }

begin
GetPixel:=ReadPixel(rp,x,y);
end;

Function  GetX;
{ -- ritorna la coordinata x corrente -- }
{ -- del cursore grafico              -- }
{ Creata: SETTEMBRE 1991                 }

begin
GetX:=CPx;
end;

Function  GetY;
{ -- ritorna la coordinata y corrente -- }
{ -- del cursore grafico.             -- }
{ Creata: SETTEMBRE 1991                 }

begin
GetY:=CPy;
end;

Function  GraphErrorMsg;
{ - Ritorna una stringa messaggio a     - }
{ - seconda di ErrCode, ritornato da    - }
{ - GraphResult                         - }
{ - NOTA: vengono ritornati solo due    - }
{ -  messaggi: "Tutto Ok", "Errore"     - }
{ -   a differenza dei messaggi TP,     - }
{ -   specifici per ogni errore.        - }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }

begin
if ErrorCode=GrOk then
    GraphErrorMsg:="TpGrafica: Tutto Ok."
  else
    GraphErrorMsg:="TpGrafica: Errore !";
end;

Procedure LineRel;
{ - Traccia una linea a partire dalla  - }
{ - posizione corrente del cursore     - }
{ - grafico fino al punto distante     - }
{ - dx sull'asse x e dy sull'asse y    - }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }

begin
{ * inizia la riga dalla posizione * }
{ * corrente del cursore grafico   * }
Move(rp,CPx,CPy);

{ * poi spostalo al termine della * }
{ * riga                          * }
CPx:=CPx+Dx;
CPy:=CPy+Dy;
Draw(rp,CPx,CPy);
end;

Procedure LineTo;
{ - Traccia una linea a partire dalla  - }
{ - posizione corrente del cursore     - }
{ - grafico fino al punto di           - }
{ - coordinate x,y                     - }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }

begin
Move(rp,CPx,CPy);
Draw(rp,x,y);
CPx:=x; CPy:=y;
end;

Procedure MoveRel;
{ -- sposta il cursore grafico di dx  -- }
{ -- sull'asse x e dy sull'asse y.    -- }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }

begin
CPx:=CPx+Dx;
CPy:=CPy+Dy;
end;

Procedure MoveTo;
{ -- muove il cursore grafico a (x,y) -- }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }

begin
CPx:=x; CPy:=y;
end;

Procedure PutPixel;
{ -- scrive un pixel alle coordinate  -- }
{ -- x,y con il colore pixel, senza   -- }
{ -- alterare il colore corrente      -- }
{ -- scelto mediante la funzione      -- }
{ -- SetColor.                        -- }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }
begin
{ - imposta la penna corrente con - }
{ - il numero di colore contenuto - }
{ - in pixel                      - }
SetAPen(rp,pixel);

{ - scrive il pixel - }
WritePixel(rp,x,y);

{ - ritorna al colore corrente di - }
{ - default, scelto con SetColor  - }
SetAPen(rp,ColoreCorrente);

end;

Procedure Rectangle;
{ - Traccia un rettangolo che ha per   - }
{ - diagonale la congiungente (x1,y1)  - }
{ - con (x2,y2)                        - }
{ -------------------------------------- }
{ Creata: SETTEMBRE 1991                  }
{ -------------------------------------- }

begin
Move(rp,x1,y1);
Draw(rp,x1,y2);
Draw(rp,x2,y2);
Draw(rp,x2,y1);
Draw(rp,x1,y1);
end;

{ ************************************** }
{ * CODICE DI STARTUP DELLA UNIT     * }
{ ************************************** }
{ Creato: LUGLIO 1991                     }
{ Modif.: SETTEMBRE 1991                  }

begin
Scr:=NIL;
Win:=NIL;
TitoloSchermo:='UnitGrafica v1.0 - by FB';

{ ** AGGIUNTO: SETTEMBRE 1991     ** }
{ -- muove il cursore grafico     -- }
{ -- alla posizione 0,0           -- }
MoveTo(0,0);
end.
```

La "nuova" Unit TPgrafica che consente di emulare, in Kick Pascal Amiga, la grafica del Turbo Pascal Borland

```
program DemoGrafico2;
{ ------------------------------------- }
{ KPDEMO2.p                             }
{                                       }
{ demo grafico                          }
{ PutPixel,LineRel,GetX,GetY,MoveTo     }
{ ------------------------------------- }
{ by Filippo Bosi                       }
{ Settembre 91 - VERSIONE KICKPASCAL    }
{ ------------------------------------- }

uses TPGrafica;

var
  i,Mx,My,Color: Word;

begin
  { * apri le librerie grafiche * }
  Gr_OpenLibs;

  { * schermo a bassa risoluzione * }
  { * con 6 piani di bit, cioe'   * }
  { * 2^6=64 colori --> Extra     * }
  { *                  HalfBrite * }
  InitGraph(GR_LORES,4);
  if GraphResult <> grOk then Halt(1);

  Mx :=GetMaxX;
  My :=GetMaxY;
  Color := GetMaxColor;
  Randomize;

  { * NOTA: nel ciclo della versione * }
  { * KickPascal del programma demo  * }
  { * viene utilizzato un for da 1 a * }
  { * 400 piuttosto che un repeat    * }
  { * until KeyPressed: tutto cio'   * }
  { * perche' la funzione KeyPressed * }
  { * di KickPascal e' messa a       * }
  { * disposizione dalla unit Crt su * }
  { * di una finestra sullo schermo  * }
  { * WorkBench. Vedremo in uno dei  * }
  { * prossimi articoli come poterla * }
  { * utilizzare                     * }

  for i:=1 to 400 do
    begin

      { * disegna un pixel colorato * }
      { * usando PutPixel           * }

PutPixel(Random(GetMaxX),Random(GetMaxY),
         Random(Color));

    SetColor(Random(Color));

    { * traccia una linea con LineRel * }
    LineRel(Random(11)-3,Random(11)-3);

    { * ...un po' di ritardo * }
    Delay(2);

    { * se abbiamo passato il bordo * }
    { * dello schermo, ritorniamo   * }
    { * all'inizio                  * }
    if (GetX>Mx) then MoveTo(0,GetY);
    if (GetY>My) then MoveTo(GetX,0);

    end;

    { * attendi che venga premuto un * }
    { * tasto oppure un bottone del  * }
    { * mouse sullo schermo grafico  * }
    Gr_AttendiUnTasto;
    { * chiudi lo schermo grafico * }
    CloseGraph;

  { * chiudi le librerie * }
  Gr_CloseLibs;
end.
```

Versione Kick Pascal Amiga del primo programma dimostrativo (in alto).
Versione Turbo Pascal dello stesso programma (in basso).

```
program DemoGrafico2;
{ ------------------------------------- }
{ TPDEMO2.p                             }
{                                       }
{ demo grafico                          }
{ PutPixel,LineRel,GetX,GetY,MoveTo     }
{ ------------------------------------- }
{ by Filippo Bosi                       }
{ Settembre 91 - VERSIONE TURBO PASCAL  }
{ ------------------------------------- }
{ N.B. per i commenti vedere la versione }
{      Kick Pascal.                     }

uses
  Crt, Graph;

var
  Gd, Gm : Integer;
  Color  : Word;
  Mx,My  : Word;

begin

  Gd := EGA64; InitGraph(Gd, Gm, '');
  if GraphResult <> grOk then Halt(1);

  Mx :=GetMaxX;
  My :=GetMaxY;

  Color := GetMaxColor;
  Randomize;

  { * ciclo principale * }
  repeat

PutPixel(Random(GetMaxX),Random(GetMaxY),
         Random(Color));
    SetColor(Random(Color));
    LineRel(Random(11)-3,Random(11)-3);
    Delay(5);

    if (GetX>Mx) then MoveTo(0,GetY);
    if (GetY>My) then MoveTo(GetX,0);

  until KeyPressed;
  ReadLn;
  CloseGraph;
end.
```

```
program Carlotta;
{ ------------------------------------- }
{ ARCS.p                                }
{                                       }
{ demo per l'utlizzo della procedure    }
{ Arc,Rectangle,GetMaxColor,SetWriteMode }
{ Circle.                               }
{ ------------------------------------- }
{ by Filippo Bosi                       }
{ Settembre 91 - VERSIONE KICKPASCAL    }
{ ------------------------------------- }

  uses TPGrafica;

  var
    ErrCode,i,x     : Integer;

  begin
    { * apri le librerie grafiche * }
    Gr_OpenLibs;

    { * schermo a bassa risoluzione * }
    { * con 6 piani di bit, cioe'   * }
    { * 2^6=64 colori --> Extra     * }
    { *                  HalfBrite * }
    InitGraph(GR_LORES,6);
    ErrCode := GraphResult;

    { * esempio di utilizzo di * }
    { * GraphErrorMsg          * }
    Writeln(GraphErrorMsg(ErrCode));

    { * se e' tutto Ok, allora disegna * }
    { * altrimenti esci dal programma  * }
    if ErrCode <> grOk then
      begin
      Gr_CloseLibs;
      halt(10); {* esci con codice 10 * }
      end;

     { * esempio di utilizzo di * }
     { * GetMaxColor            * }
     Writeln('N.colore massimo:',
             GetMaxColor);

     { * disegna archi e rettangoli * }
        for i:=1 to GetMaxColor do
         begin
         SetColor(i);
         SetWriteMode(NormalPut);
         Arc(100,100,10+i*3,
           , 300-i*3,130-i*2);
         Rectangle(30+i*3,10,100+i*3,100);
         end;

         { * ...un po' di cerchi * }
         for i:=1 to 100 do
          begin
          SetColor(i mod GetMaxColor);
          Circle(GetMaxX,GetMaxY,80+i);
          end;

{ * esempio di "animazione" usando il * }
{ * modo XORPut: se riscriviamo lo    * }
{ * stesso oggetto su video e' come   * }
{ * se non avessimo eseguito alcuna   * }
{ * operazione sullo schermo grafico  * }
{ * che fa cosi' da "sfondo" al       * }
{ * nostro rettangolino che viaggia   * }
{ * per lo schermo                    * }

        SetWriteMode(XORPut);
        for x:=1 to GetMaxY do
          begin
          Rectangle(x,x,x+20,x+20);
          Delay(1);
           { * un po' di ritardo per  * }
           { * limitare il flickering * }
          Rectangle(x,x,x+20,x+20);
          end;
         SetWriteMode(NormalPut);

        { * attendi che venga premuto un * }
        { * tasto oppure un bottone del  * }
        { * mouse sullo schermo grafico  * }
        Gr_AttendiUnTasto;

        { * chiudi lo schermo grafico * }
        CloseGraph;
      { * chiudi le librerie * }
      Gr_CloseLibs;
end.
```

Versione Kick Pascal Amiga del secondo programma dimostrativo

Prima di iniziare a lavorare con un microprocessore (soprattutto quando questo è un 68000...) è bene informarsi sul suo modo di ragionare; o meglio, sui tanti suoi modi di elaborare i dati

Tutti gli indirizzamenti del 68000

< di Luigi Callegari>

< Esaminare C. C. n. 86 >

Iniziamo, da questo mese, numerose trattazioni illustrative sulla programmazione dei microprocessori della famiglia **68000**, cuori dei computer **Amiga**, che avrà un taglio pratico molto simile al corso per 80X86 tenuto sulle pagine di Computer Club da Giancarlo Mariani.

Il microprocessore 68000 dispone di un set di istruzioni non particolarmente ampio, ma estremamente potente e flessibile e, soprattutto, ricco di numerosi modi di indirizzamento. Ciò significa che la maggior parte delle istruzioni del 68000 possono accettare i loro operandi in una varietà di forme. Gli operandi possono infatti essere gli stessi **registri**, le **locazioni** di memoria indirizzate in vari **modi**, perfino **inclusi** nella istruzione stessa. Per "coincidenza", l'istruzione più usata del microprocessore 68000 è MOVE, che prevede praticamente tutti i modi di indirizzamento. Descrivendola è quindi possibile illustrare, con sufficiente completezza, i vari modi di indirizzamento del processore, comuni a tutte le altre istruzioni del che saranno trattate prossimamente. Vedremo ora, riferendoci alla figura ed alla tabella dei modi di indirizzamento, di fare un esempio per ciascuno di questi.

MOVE

MOVE	<ea1>,	<ea2>

Nella sintassi Motorola, il codice mnemonico MOVE significa, in inglese, MUOVI.

Il suo scopo, infatti, è quello di trasferire il contenuto di registri interni o di locazioni di memoria.

Il simbolo **<ea>** significa, nella documentazione ufficiale, **effective address**, ovvero **indirizzo effettivo**.

In pratica, questa istruzione copia il contenuto della **locazione** di memoria o del **registro** interno indicato da **ea1** nella locazione di memoria o registro interno indicato da **ea2**.

Nel set 68000, il primo operando si chiama **sorgente**, il secondo **destinazione**, dato che il primo indica ***dove*** andare a prendere il dato o ***quale*** è il dato su cui svolgere l'operazione indicata dallo mnemonico, il secondo operando ***dove*** memorizzarlo.

Modi di indirizzamento validi:

Sintassi	Sorgen.	Destin.
Dn	X	X
An	-	-
(An)	X	X
(An)+	X	X
-(An)	X	X
(d16,An)	X	X
(d8,An,Xn)	X	X
(bd,An Xn)	X	X
xxx.W	X	X
xxx.L	X	X
#<dato>	X	-
(d16,PC)	X	-
(d8, PC, Xn)	X	-
(bd, PC, Xn)	X	-

Indirizzamento diretto da registro

I dati che costituiscono gli operandi sono contenuti in uno dei registri dati o dei registri di indirizzamento.
Il nome di ***registro dato*** si scrive come **Dn** e quello di un ***registro indice*** con **An**, dove "**n**" vale ovviamente tra **0** e **7**, ad indicare i sedici registri a 32 bit del 68000. Ad esempio...

MOVE.L A7, D3

...copia tutti i 32 bit contenuti nel registro **A7** nel registro **D3** (per la spiegazione della "**.L**", indicante la *dimensione* degli operandi, vedere il riquadro specifico). Si ricordi che il registro **A7** è particolare, dato che è usato come Stack Pointer dal 68000.

La **sorgente** può essere un registro indice o dati, la **destinazione** può essere soltanto un registro indice (vedere oltre l'istruzione MOVEA per specificare come destinazione un registro indice).

Indir. indirett. da registro

Ci sono 5 modi di indirizzamento su un indirizzo-operando contenuto in uno dei registri indice (**A0 — A7**). L'**indirizzamento indiretto** è così chiamato perchè l'operando non è il registro stesso, bensì il valore nella locazione di memoria puntata dal registro stesso. Ad esempio...
MOVE.L #0, (A2)
...pone a zero i trentadue bit di memoria a partire dalla locazione di memoria contenuta nel registro indice **A2**. Attenzione: se **An** contiene un indirizzo dispari, le operazioni su **W** e **L** (word e longword) danno una Guru Meditation col 68000 (i sistemi con 68020/30 sono più tolleranti). Analogamente...
MOVE.W (A0),D1
...pone in **D1** la word collocata all'indirizzo indicato dal contenuto del registro **A0**.

Indirizzamento assoluto

In questo caso l'**operando** in memoria viene indicato tramite il suo **indirizzo**. In altre parole, l'**indirizzo effettivo** stesso (<ea>) viene specificato quale operando. Il 68000 ha due modi di indirizzamento assoluto: indirizzamento **assoluto corto** (con operando a **16** bit, esteso a 32 bit) e indirizzamento **assoluto lungo** (operando a **32** bit).
Il ***primo*** modo può essere usato solo per indicare indirizzi di locazioni di memoria dei **primi 32K** ($000000... $007FFF) o negli **ultimi 32K** ($FF8000... $FFFFFF) della mappa di Amiga, mentre il ***secondo*** modo può ovviamente indicare qualunque locazione nei 16M indirizzabili dal 68000. Le istruzioni indirizzate col modo assoluto corto occupano una parola in meno e sono eseguite più velocemente. Ad esempio:

```
MOVE.L Buffer,  D0
MOVE.W $3F00,   D1
MOVE.W $03F00,  D1
```

La prima istruzione scrive nel registro **D0** i 32 bit (longword) trovati alla locazione stabilita dall'etichetta Buffer, la seconda scrive nel registro **D1** i sedici bit contenuti alla locazione $3F00 (l'assembler usa automaticamente l'indirizzamento **corto**, dato che l'indirizzo è raffigurabile con 16 bit) mentre la terza è identica ma costringe l'assemblatore ad usare l'indirizzamento assoluto **lungo** (il numero è visto come una cifra a 32 bit a causa dello zero iniziale che ne forza l'interpretazione da parte dell'Assembler come Longword).
Il risultato è che l'istruzione generata dalla seconda sintassi è composta da 2 parole e richiede **12 cicli di clock** del processore per essere eseguita; l'istruzione generata dalla terza istruzione occupa invece 3 parole e richiede **16 cicli di clock** (sul processore 68000, i fratelli maggiori possono avere altre temporizzazioni).
Un altro esempio:

MOVE.L $FFFE.W, D4

pone in **D4** la longword (MOVE.L, notare) collocata all'indirizzo $FFFFFE. L'indirizzo sorgente, come detto, viene calcolato tramite "estensione del segno"; vedere il riquadro per ulteriori approfondimenti circa questa particolarità del 68000.

Indirizzamento da registro più scostamen.

L'indirizzo contenuto in un **registro indice** viene usato insieme ad un valore, detto **scostamento** ("**offset**"), che è un numero a 16 bit con segno.
Ad esempio...
MOVE.W 22 (A0), D1
...pone nel registro **D1** la word collocata all'indirizzo ricavato sommando algebricamente **22** al contenuto del registro indice **A0**.
Poichè lo scostamento è calcolato come un numero intero con segno a 16 bit, questo modo di indirizzamento può raggiungere i **32767** byte più in alto od i **32768** byte più in basso in memoria, rispetto all'indirizzo nel registro indice.

Indiretto da registro con scostamento ed indice

Simile al "registro più scostamento", può aggiungere il contenuto di un altro registro, avendo valore di scostamento un numero di soli 8 bit:

MOVE.W 12 (A0, D0.W), D1
...pone nel registro **D1** la word all'indirizzo calcolato sommando algebricamente **12** al contenuto del registro **A0** (32 bit) ed alla word contenuta nel registro **D0**. Si può ovviamente usare anche qualcosa come...
MOVE.W michela (A1, D1.L), D2
...per porre nel registro **D2** il valore calcolato sommando algebricamente il valore della etichetta **michela** al contenuto del registro indice di base **A1** ed i 32 bit (**.L**) del registro **D1**.

Indirizzamento relativo

Il 68000 prevede due modi per indirizzare la memoria relativamente al valore del registro **Program Counter** (PC). Di solito si sfrutta questa possibilità per i salti nel codice rilocabile, ma è pure conveniente utilizzarla per accedere a valori costanti nel programma. Non è però possibile scrivere in una locazione indirizzata in questo modo e ciò incoraggia la stesura di "codice puro", ovvero di codice che non si automodifica durante l'esecuzione (particolarmente scoraggiato in ambiente Amiga), perciò può essere eseguito in qualunque momento da qualunque programma. Il codice assembly di questo tipo si chiama anche **codice rientrante**.
Il primo modo di indirizzamento relativo al PC si chiama **indirizzamento relativo al PC con scostamento (*offset*)**.
Ad esempio...
MOVE.W label(PC), D1

...trasferisce nel registro **D1** i sedici bit meno significativi contenuti nella locazione di memoria indicata dall'etichetta "label", calcolandone però la posizione relativamente al Program Counter.
In pratica, l'assembler calcola lo scostamento (differenza di valori assoluti) in byte tra la parola di estensione della istruzione di trasferimento e la locazione marcata dall'etichetta **label** e lo memorizza nella parola d'estensione stessa dell'istruzione.

Al momento dell'esecuzione, il 68000 carica il contenuto della locazione *label* nella word inferiore del registro D1.
Dal momento che lo scostamento è un intero a sedici bit, "label" non può superare in memoria le **16383** parole più alte rispetto alla parola di estensione dell'istruzione, nè essere inferiore alle **16384** parole più basse. Una variante di questo modo è l'**indirizzamento relativo al PC con scostamento ed indice**:

MOVE.W lidia(D0.L), D1

...farà sì che l'assembler calcoli lo scostamento tra la parola di estensione della istruzione e la locazione con etichetta **lidia**, memorizzandola nella parola di estesione dell'istruzione.
Al momento dell'esecuzione, il 68000 aggiungerà il valore a 32 bit di **D0** all'indirizzo calcolato di partenza della tabella dati, quindi caricherà il contenuto a 16 bit della locazione di memoria indirizzata dal risultato, nei 16 bit di ordine inferiore del registro dati D1.
Dato che lo scostamento è un intero a 8 bit dotato di segno, *lidia* non deve materialmente superare le **63 word** di memoria verso l'alto o le 64 verso il basso come distanza relativa.

Indir. da PC con scost. e con scostam. ed indice

Si tratta di una replica dei precedenti, ma al posto di un registro indice qualunque, si usa **il program counter** (PC). Vi è una sola differenza, molto importante però: l'operando di destinazione non può **mai** essere scritto, ovvero tale modo di indirizzamento viene usato solo per istruzioni di salto o diramazione (che vedremo più avanti) e non per istruzioni quali MOVE.
Questa limitazione è per scoraggiare la scrittura di codice automodificante.

Indirizz. dati immediato

E' il modo più semplice, dato che il valore dell'operando è memorizzato nel **corpo** dell'istruzione stessa, essendo già noto all'assemblatore al momento della compilazione.
Si osservino il seguente esempio...
MOVE.B #$FF, D0

...ed anche questo...
MOVE.L #$12345678, D2
La prima istruzione pone il numero esadecimale $FF negli otto bit *inferiori* di **D0**, la seconda pone il numero $12345678.

Indir. da registro con predecremento o postincremento

Per semplificare la gestione dello **Stack** e la scrittura di compilatori C (!), il 68000 offre delle istruzioni molto potenti:

```
MOVE.W (A0)+, (A1) +
MOVE.L -(A2), -(A4)
```

La prima trasferisce la word contenuta all'indirizzo di memoria espresso nel registro **A0** nella locazione indicata dal contenuto di **A1**; *in seguito* incrementa di una "word" (cioè la stessa *dimensione* dell'operando spostato) ambedue i registri.
La seconda istruzione, analogamente, *dapprima* decrementa di una longword i valori di A0 e A1, *poi* trasferisce la longword contenuta nella locazione puntata da **A2** in quella puntata da **A4**.

Si noti che il "prima" ed il "poi" dell'incremento o del decremento dipendono dalla posizione dei segni **più** (+) e **meno** (—) rispetto all'indice posto tra parentesi. Si può anche usare...

```
MOVE.L  D1,      - (A0)
MOVE.B  (A6)+,   D5
```

...per trasferire i 32 bit del registro **D1** nella locazione di memoria calcolata sottraendo una longword al registro indice **A0** (che viene effettivamente decrementato prima del trasferimento), e nel secondo caso, per trasferire il byte alla locazione di memoria indicato sommando un byte (otto bit) al valore contenuto di **A6** e *poi* trasferendolo il byte nell'indirizzo così calcolato nel registro dati **D5**.

Pensando al poi...

Prossimamente descriveremo sistematicamente tutte le istruzioni del 68000, raggruppate per tipo.

Dimensioni degli operandi

I processori della famiglia 68000 sono a 32 bit **effettivi** *internamente*, dato che possono eseguire (quasi) tutte le istruzioni con una dimensione di 32 bit.

I registri **interni**, ad eccezione del Program Counter nel 68000/10, sono a **32** bit, ma il programmatore può specificare su quale dimensione di dato sta lavorando: byte (**B**), word (**W**) oppure longword (**L**).
Per convenzione, il **byte** è un gruppo di 8 **bit**, la **word** è un gruppo di 16 bit e la **longword** è un gruppo di 32 bit. Ad esempio, l'istruzione...
MOVE.L D4, D5

...copia tutti i 32 bit contenuti nel registro **D4** nel registro **D5**. Invece...

MOVE.B D3, D2

...copia soltanto gli otto bit "più bassi" del registro **D3** nel registro **D2**. In pratica, in quest'ultimo i rimanenti 24 bit compresi tra il numero 8 ed il numero 32 rimangono immutati (non vengono azzerati). Analogamente, le istruzioni...

MOVE D7, D0
MOVE.W D7, D0

...sono, per tutti gli assembler attualmente diffusi per Amiga, la stessa identica cosa (la "dimensione" per default è "word") e ricopiano *soltanto* i sedici bit inferiori (da bit 0 a bit 15) da D7 nei sedici bit inferiori di D0, lasciando inalterati i bit da 16 a 31 in D0.

Convenzioni di memorizzazione

Ogni processore ha dei modi personali non solo di esprimere nelle istruzioni gli operandi sorgente e destinazione, ma anche di riporre i dati nei registri od in memoria.

Ad esempio, l'istruzione...
MOV DX, SP
...dei processori 80X86 **copia** il contenuto dello Stack Pointer (**sorgente**) nel registro DX (**destinazione**), mentre la corrispondente in assembler 68000 è...
MOVE SP, D0
...dato che la sorgente (**SP**, ovvero **A7**) deve essere specificata sulla linea *prima* della destinazione (**D0**). Per facilitare le cose, con il 68000 si immagini che tra gli operandi, invece di una virgola, vi sia un segno di uguaglianza, cioè **SP = D0**.
Analogamente, bisogna ricordare che il processore 68000 scrive in RAM prima il byte più significativo di una word, poi il byte meno significativo. In altre parole, scrivendo una word od una longword (cioè due word), il byte meno significativo occupa sempre la posizione in memoria più alta.
Inoltre, bisogna ricordare che col 68000 l'accesso ad operandi in formato byte può avvenire sia ad indirizzi **pari** che ad indirizzi **dispari**.
Al contrario, l'accesso ad operandi in formato word o longword deve *sempre* avvenire ad indirizzi **pari** (pena Guru Meditation).
I processori **68020** e **68030** hanno limitazioni diverse che, quando necessario, saranno precisate.

Estensione del segno

L'estensione del segno è il procedimento con il quale *si trasforma* tipicamente un valore a **sedici** bit in uno a **trentadue** bit, rispettando le regole della matematica binaria.
Si tratta di un concetto usato genericamente nella programmazione Assembler, ma che riveste particolare importanza con i processori della famiglia 68000, per la loro capacità di usare indifferentemente formati di dati "misti" per esprimere gli operandi: **byte**, **word** e **longword**.
In pratica, con l'estensione del segno di un numero a 16 bit, per ottenere un numero a 32 bit si impone che il bit quindici del numero di partenza venga ricopiato in tutti i sedici bit superiori del numero a 32 bit da formare.
Nel set di istruzioni del processore 68000 esiste una funzione apposita per svolgere esplicitamente questa operazione, chiamata EXT, ma bisogna ricordare che tale estensione viene spesso eseguita automaticamente dal microprocessore, come ad esempio nel caso di indirizzamento assoluto corto.
Ad esempio, supponiamo di avere il numero esadecimale $8FFE, che in binario si scrive:
```
1000 1111 1111 1110
```
Estendendolo a 32 bit, si ottiene:
```
1111 1111 1111 1111
1000 1111 1111 1110
```
Ovvero, dato che il quindicesimo bit è **uno**, i sedici bit meno significativi del numero $8FFE esteso a 32 bit devono essere degli uni.
Il risultato è il numero visto, che in esadecimale è $FFFF8FFE. Se, invece, il numero fosse ad otto bit, ad esempio...

```
0101 1111
```

...in esadecimale $5F, convertendolo con estensione di segno a sedici bit si otterrebbe il numero:

```
0000 0000 0101 1111
```

Ovvero ancora $005F.

Amiga si affaccia alla finestra

< di Luigi Callegari>

< Esaminare articoli sul "C" sui n. 82, 83, 84, 85, 86 >

Il sistema operativo di Amiga è strutturato secondo **librerie**, ovvero collezioni di routine, descritte ampiamente nella documentazione ufficiale redatta dalla Commodore, i famosi *Rom Kernel Manuals* già più volte citati in numeri precedenti.

Uno degli aspetti più appariscenti di Amiga è l'**interfaccia utente**, che si basa sullo sfruttamento dei vari **gadget, menu, mouse** tramite apposite funzioni della libreria chiamata **Intuition**, scritta da *Robert Mical* in linguaggio C.

Lo sfruttamento di tale interfaccia grafica non è obbligatoria, basti pensare a tutti i comandi eseguibili solo da CLI, ma consente comunque di sviluppare programmi professionali in maniera molto più semplice ed efficiente per l'utente.

Inoltre l'uso delle funzioni della libreria Intuition sono strettamente connesse con quelle della **libreria Graphics**, e tutti sappiamo quanto sia preziosa la grafica per qualunque computer...

Lo schermo

L'unità base di visualizzazione di Intuition è lo schermo, in inglese **Screen**. Intuition può lavorare contemporaneamente su vari schermi, ma ciascuno di essi deve avere effettivamente un'ampiezza orizzontale sempre massima (**640** punti in modo interlacciato oppure **320** punti in modo normale).

Uno schermo è materialmente una zona della memoria CHIP contenente i dati specificanti, in modo simbolico, ciò che deve formare la rappresentazione sullo schermo del monitor.

Ciascuno schermo è formato da uno o più **bitplanes**, materialmente delle sequenze di bit in CHIP RAM, che indicano lo stato dei pixel secondo un particolare codice.

In effetti, il modo in cui i bitplanes vengono interpretati dal sistema operativo di Amiga dipendono anche dal "modo" grafico selezionato: HAM, EXTRA_HALFBRITE eccetera.

In bassa risoluzione abbiamo **320** pixel elementari per ogni linea dello schermo video per un massimo di **256** linee in modo PAL non interlacciato e **512** linee per il modo PAL interlacciato.

Quest'utimo è proprio quello che provoca lo sfarfallio delle immagini sul video se non si possiede un Amiga 3000 con monitor adeguato od il VDE sugli altri Amiga.

Bitplanes e colori

Quando si usa **un solo** bitplane, sullo schermo possiamo avere ovviamente un massimo di **due colori**: dal momento che ciascuno dei pixel sullo schermo può corrispondere ad un solo bit in memoria, il quale a sua volta può assumere solo due valori binari (**0** od **1**), i colori indicabili sono solo due, ad esempio il bianco ed il nero.

Volendo usare più colori bisogna usare più bitplanes, ovvero più CHIP RAM per contenerne i dati di definizione. I limiti attuali dell'hardware degli Amiga 500/2000 con Kickstart e Workbench **V1.3** prevedono che si possano avere non più di **5 bitplane** (**32** colori) in bassa risoluzione e **4 bitplane** (**16** colori) in alta risoluzione. Ovviamente, per ora, parliamo sempre e soltanto dei modi grafici standard, non dei particolari HAM ed EXTRA_HALFBRITE.

Usando più bitplanes l'hardware di Amiga legge, per ciascun pixel dello schermo, da uno a cinque bit e forma un numero di identificazione per il colore da produrre.

Ad esempio, sapendo che il primo punto dello schermo è controllato dal primo bit di ciascuno dei **3** bitplanes (**8** colori al massimo) e volendo attribuire al punto il

I PICCOLI COMMODORE SONO CRESCIUTI

Struttura NewScreen

Ecco la definizione della struttura **NewScreen**, che definisce per la funzione **OpenScreen()** lo schermo da aprire, come riportata anche nel file di inclusione screens.h della directory intuition...

```
struct NewScreen {
   SHORT   LeftEdge, TopEdge;
   SHORT   Width, Height, Depth;
   UBYTE   DetailPen, BlockPen;
   USHORT  ViewModes, Type;
   struct TextAttr *Font;
   UBYTE   *DefaultTitle;
   struct Gadget *Gadgets;
   struct Bitmap *CustomBitMap;
}
```

...dove i parametri hanno il seguente significato:

LeftEdge	Coordinata X spigolo super. sinistro (0)
TopEdge	Coordinata Y linea inizio scansione
Width	Larghezza (320 o 640)
Height	Altezza
Depth	Numero di bitplanes (1...6)
DetailPen	Registor colore per i dettagli
BlockPen	Registro colore per le aree
ViewModes	Modo di presentazione
Type	Tipo dello schermo
Font	Puntatore a struttura definizione testo
DefaultTitle	Puntatore a stringa intestazione
gadgets	Puntatore a primo gadget
CustomBitMap	Puntatore a bitmap personale

colore numero **3**, si assegneranno i valori binari (**0 - 1 - 1**) ai primi bit del primo, secondo e terzo bitplane rispettivamemte.

Al momento della creazione, da parte dell'hardware, del quadro video da realizzare sul tubo catodico, l'hardware di Amiga fornirà i segnali necessari per indicare di accendere, nella posizione corrispondente al primo pixel, un punto avente il colore numero 3.

Il numero identificatore del colore ricavato dai bitplanes è effettivamente un riferimento ad uno dei **32** (al massimo) registri hardware interni al chip custom **Denise**, che contiene fisicamente la codifica del colore da usare.

Questo è conservato nel registro hardware come un numero a **12** bit che specifica le dosi di **Rosso**, **Verde** e **Blu** (**R**ed, **G**reen, **Bl**u, i tre colori fondamentali dei sistemi ad emissione di luce, che mescolati consentono di ottenere l'intero spettro ottico) da usare per ottenere una qualunque delle **4096** sfumature di colore disponibili su Amiga.

Il numero 4096 deriva dal fatto che la composizione del colore si basa su 3 componenti cromatiche (RGB), ciascuna controllata da **4** bit, quindi con **16** varianti per un totale di 16 x 16 x 16 (Rosso x Verde x Blu =) 4096 combinazioni, appunto.

In ambiente Amiga non si parla tanto di *"registri colore"*, bensì di penne (**Pen** in inglese), in quanto dal punto di vista del software si usano dei numeri (da **0** a **31**) corrispondenti al contenuto dell'omonimo registro interno di Denise.

Funzioni di base

Sebbene molti programmi utilizzino direttamente lo schermo fornito dal Workbench, in risoluzione 640 x 256, spesso può rendersi necessario disporre di una risoluzione diversa o di uno schermo personalizzato, ad esempio con una *tavolozza* di colori differente da quella del Workbench standard.

La funzione **OpenScreen()** serve appunto ad aprire uno schermo. Essa accetta come input un puntatore ad una struttura **NewScreen** (vedere riquadro) che indica, appunto, le caratteristiche dello schermo da aprire.

La funzione restituisce un puntatore ad una struttura Screen che contiene fisicamente i dati di gestione "dinamica", ovvero tutti quei dati che Intuition deve potere regolare durante il funzionamento di un nostro programma.

Tale puntatore vale NULL nel caso l'apertura non sia riuscita: il nostro programma deve, ovviamente, controllare questa condizione di rientro per prendere eventuali provvedimenti.

La complementare di OpenScreen() è **CloseScreen()**, che accetta come parametro il puntatore a struttura Screen restituito da OpenScreen() ed esegue la chiusura dello schermo, cancellando dalla memoria CHIP la struttura Window di controllo dinamico e restituendo questa memoria al sistema.

Struttura NewScreen

Nel riquadro abbiamo riportato la definizione della struttura NewScreen che bisogna definire nei nostri programmi C per specificare le caratteristiche dello schermo da aprire.

Date le limitazioni degli schermi prima dette, è ovvio che il valore per la coordinata **X** dello spigolo superiore sinistro (**LeftEdge**) è sempre zero.

Il campo **ViewModes** contiene una costante, od una somma di costanti ottenuta con gli operatori di somma (**+**) oppure **OR** logico (|), tra le varie disponibili.

Tali definizioni sono rintracciabili nel file `graphics/view.h`:

`NULL`	`Bassa ris. (320/352 pixel)`
`HIRES`	`Alta ris. (640/704 pixel)`
`LACE`	`Interlacciato (512 linee)`
`SPRITES`	`Abilitazione sprites`
`DUALPF`	`Doppio campo`
`HAM`	`Modo con 4096 colori`
`EXTRA_HALFBRITE`	`Modo lumin. dimezz.`

Il modo interlacciato è quello che provoca lo sfarfallio se non si possiede il VDE ed un adeguato monitor.

Gli **sprites** sono *oggetti grafici* definiti via software e gestiti anche per mezzo dell'hardware speciale di Amiga, già ben noti anche ai programmatori del Commodore 64.

Il modo speciale **Dual Playfield** è piuttosto particolare ed usato raramente, consentendo la visualizzazione di due schermi sovrapposti in modo che il colore zero del primo schermo risulti trasparen-

La struttura NewWindow

Ecco la definizione della struttura **NewWindow**, usata dalla funzione **OpenWindow()** della libreria Intuition per aprire una finestra, riportata nel file **intuition.h** della directory di inclusione intuition:

```
struct NewWindow {
SHORT LeftEdge, TopEdge;
SHORT   Width, Height;
UBYTE   DetailPen, BlockPen;
ULONG   IDCMPFlags;
ULONG   Flags;
struct Gadget*  FirstGadget;
struct Image *CheckMark;
UBYTE   *Title;
struct Screen    *Screen;
struct BitMap    *BitMap;
SHORT   MinWidth, MinHeight;
USHORT  MaxWidth, MaxHeight;
USHORT Type;
};
```

dove i parametri sono:

LeftEdge	Coordinata X di inizio finestra sullo schermo
TopEdge	Coordinata Y di inizio finestra sullo schermo
Width	Larghezza in pixel della finestra
Height	Altezza in pixel della finestra
DetailPen	Penna per barra e gadget
BlockPen	Penna per blocchi e contorni
IDCMPFlags	Flag di controllo per la IDCMP
Flags	Flag di controllo caratteristiche e gadget
FirstGadget	Punta al primo gadget personalizzato
CheckMark	Punta a definizione di spunta personalizzata
Title	Punta a stringa di intestazione
Screen	Punta ad eventuale schermo custom definito
BitMap	Punta ad eventuale Superbitmap personale
MinWidth	Minima larghezza nel ridimensionamento
MinHeight	Minima altezza nel ridimensionamento
MaxWidth	Massima larghezza nel ridimensionamento
MinWidth	Minima larghezza nel ridimensionamento
Type	Tipo di schermo

te e consenta la visualizzazione del secondo schermo "sottostante"; tale modo consente al massimo **8** colori.

Il famoso modo HAM lavora solo in bassa risoluzione, 320 pixel verticali o 352 in overscan, ma consente di avere sino a 4096 colori contemporaneamente sullo schermo con alcune limitazioni tra i colori adiacenti in una stessa riga orizzontale. Si usano **6 bitplane**, ma 4 sono usati per indicare 16 colori di base ed altri 2 bitplane per codificare ciascun pixel in funzione del precedente.

Il modo HALFBRITE consente sino a 64 colori in bassa risoluzione e richiede 6 bitplanes. I primi 5 bit di ogni pixel definiscono il colore specificando quale dei 32 registri di colore di Denise bisogna leggere per produrlo, mentre il **sesto bit** indica se esso va reso *normalmente* o a *luminosità dimezzata*. E' evidente che, nel secondo gruppo, i 32 colori sono limitati ad essere identici ai primi 32, ma con luminosità dimezzata.

Il campo **Type** definisce il tipo di schermo voluto. Di regola, si specifica sempre CUSTOMSCREEN, mentre se si desidera specificare una propria bitmap personale per lo schermo si deve aggiungere anche il flag CUSTOMBITMAP.

Il campo **Font** contiene un puntatore ad una struttura **TextAttr**, di cui parleremo prossimamente, che definisce il set di caratteri da usare. Se si specifica NULL, viene usata la fonte di default. Anche dei gadget parleremo in seguito, essendo un argomento molto lungo e complesso.

Apriamo uno schermo

Il primo listato illustra praticamente come definire uno schermo personalizzato (*custom*) e ad aprirlo, cioè a visualizzarlo materialmente.

Si inizia con l'inclusione del file **intuition.h** della directory **Intuition**. Se si legge questo file, si scopre che include, con altre direttive, parecchi file di varie directory.

Nei primi semplici programmi per Intuition, molto spesso non occorrerebbe includere null'altro, ma abbiamo inserito, per raffinatezza, l'inclusione dei file di prototipo se si usa **Lattice/SAS** o **Aztec C V5**.

Parleremo in seguito diffusamente dei **File Di Prototipo**; per ora basti sapere che consentono essenzialmente al compilatore (1) di eseguire **controlli più rigidi** sui parametri passati ed il modo in cui si usano le funzioni di libreria, (2) di generare **codice più veloce** tramite chiamata diretta delle funzioni di libreria (senza passare per codice di interfaccia letto ed inserito in fase di linking dal file amiga.lib per SAS/C). Scopo simile ha il singolo file **functions.h** fornito con Aztec C.

Si prosegue con la definizione di due puntatori globali: IntuitionBase e schermo. La loro necessità è concettualmente piuttosto diversa.

IntuitionBase è un nome riservato usato dal compilatore per indirizzare tutte le funzioni della libreria Intuition. I compilatori Amiga assumono che in esso sia contenuto l'indirizzo iniziale (di "base") a cui inizia la libreria di funzioni di Intuition. Se il valore qui inserito, restituito da una opportuna chiamata alla funzione OpenLibrary() di Exec, è nullo od errato ed il programma non termina immediatamente, quando si tenterà di usare una qualunque funzione di Intuition si otterrà una Guru Meditation.

Difatti, se si guarda nella funzione **_main()**, si vede che il programma tenta di aprire la libreria "intuition.library" tramite la funzione OpenLibrary() di Exec, che restituisce appunto l'indirizzo di base della libreria. Se tale funzione restituisce un valore pari a NULL (invece dell'indirizzo base della libreria), il programma invoca la funzione di uscita ordinata **die()** restituendo al CLI, che ha chiamato il nostro programma compilato, un valore di **10**.

La funzione OpenLibrary() di Exec ha due parametri: il primo è il *nome della libreria da aprire*, il secondo è il *numero minimo di versione* della libreria richiesto. Il nome deve essere noto al programmatore, il numero di versione vale solita-

mento **0** se va bene qualunque versione della libreria (perchè non si usano funzioni particolari aggiunte rispetto alla 1.2/1.3), oppure almeno 33L se si desidera esplicitamente che la versione di Kickstart sia almeno la V1.2.

Il puntatore globale "schermo" è destinato invece a contenere l'indirizzo in memoria della struttura di controllo dinamico dello schermo aperto tramite una chiamata a funzione OpenScreen(). Molti programmi non hanno bisogno di usare tali dati, ma in ogni caso bisogna sapere che quando si invoca OpenScreen(), questa restituisce NULL se non è possibile aprire lo schermo secondo le direttive date nella struttura di definizione NewWindow (nel nostro caso chiamata **strutscher**), oppure l'indirizzo di base di una struttura di tipo Screen (definita nel file di inclusione **intuition/screen.h**) dove Intuition colloca durante l'esecuzione del programma ed il funzionamento dello schermo tutti i dati necessari alla gestione.

Come detto, OpenScreen() ha bisogno come parametro l'indirizzo di una struttura NewScreen di definizione dello schermo; difatti si noti che la chiamata è:

```
OpenScreen(&strutscher);
```

...dove la **&** prima di un nome di struttura, ricordiamo, indica l'indirizzo di partenza in memoria della struttura stessa.

Si noti che il programma esegue con un **If** il controllo sulla validità dell'apertura dello schermo. Se il valore memorizzato nella variabile **schermo** è NULL, il programma esce tramite la funzione die() restituendo un codice di errore **11** al processo CLI che ha invocato il nostro programma compilato (utilizzabile, ad esempio, per influenzare l'esecuzione di file batch con le istruzioni If, Warn, Error, eccetera).

La funzione die() si preoccupa di verificare se la libreria IntuitionBase è stata aperta, dato che può essere invocata anche nel (remoto) caso che non si riesca ad aprire questa libreria per terminare, e di chiuderla in caso positivo e, analogamente, verifica se lo schermo è stato aperto prima di chiuderlo con CloseScreen().

Questo, tra l'altro, è il motivo per cui la variabile che contiene l'indirizzo dello schermo è stata dichiarata come globale, quindi visibile a tutte le funzioni (die() compreso) e non dinamicamente in main().

Si noti anche che, in generale, tentare di "chiudere" un oggetto (libreria, schermo, finestra, device...) precedentemente non aperto significa quasi certamente provocare una Guru Meditation. E' questo il motivo per cui è buona regola garantire che si sta chiudendo qualcosa di effettivamente aperto usando un **If**, in generale, nei programmi.

Il programma termina usando la funzione di libreria AmigaDOS **Delay(ritardo)**, che impone un'attesa di "ritardo" cinquantesimi di secondo prima di proseguire l'esecuzione. Nel nostro caso, attende cinque secondi per poi fare sfociare l'esecuzione nella chiamata a die() con parametro zero per rientrare al CLI con la condizione tipica di "esecuzione completata senza errori".

Piccoli trucchi

Pur in un listato così corto, sono stati usati alcuni trucchetti che solitamente vengono spiegati agli **utenti già esperti**.

A parte l'uso dei file di prototipo già accennato, si noti la denominazione delle funzioni **main()** ed **exit()** precedute da un simbolo di sottolineatura (_).

In generale, si usano le funzioni standard main() ed exit() senza *underscore*, ad indicare quelle standard della libreria. Usando quelle precedute da underscore, il linker inserirà del codice di entrata ed uscita dal programma più corto, ovvero che esegue alcuni controlli ed operazioni in meno.

Come regola generale (ma bisogna essere cauti), se il nostro programma **non** usa allocazioni di memoria dinamiche con **malloc()**, non usa lo **I/O** standard nè file, si possono usare tali funzioni abbreviate, come nel nostro semplice programma di esempio.

Ritorneremo comunque in futuro su tale argomento, quando parleremo del codice inserito dal linker, in modo trasparente per il programmatore, nei programmi eseguibili sviluppati in C.

La finestra

In un ambiente multitasking devono essere rese resi possibili sia lo svolgimento contemporaneo di operazioni di I/O sullo schermo effettuato da più programmi, sia l'invio da parte dello stesso task dei propri output su differenti porzioni dello schermo, il che può rendere più semplice ed organico l'uso del software applicativo.

Intuition consente appunto di suddividere ciascuno schermo in più **finestre**, in inglese **windows**, le quali sono materialmente delle aree di visualizzazione dotate di particolari caratteristiche gestite in modo automatico e flessibile da parte del sistema operativo e funzioni di libreria, come ad esempio la sovrapposizione dell'una sull'altra (**overlapping**) o la traslazione sullo schermo effettuata mediante il mouse (**dragging**).

Intuition usa per gestire le finestre un'altra libreria di funzioni, chiamata **layers.library**, che comprende alcune funzioni di "basso livello" che operano su porzioni rettangolari di bitmap in memoria. Tale libreria è però raramente usata, dato che risulta più semplice ed efficace sfruttarla indirettamente tramite Intuition.

Struttura NewWindow

La gestione delle finestre è abbastanza simile a quella degli schermi.

Ad OpenScreen() e CloseScreen() fanno da complementari **OpenWindow()** e **CloseWindow()**, oltre a parecchie altre funzioni di supporto delle finestre di cui parleremo nelle prossime (numerose) puntate dedicate ad Intuition.

Si noti che nella definizione di una finestra la larghezza e l'altezza massime consentite, specificate nei campi **Width** ed **Height** di **NewWindow**, dipendono ovviamente dalla risoluzione orizzontale e verticale dello schermo di appartenenza e dalle coordinate di inizio della finestra stessa, ovvero normalmente:

Width = 640 - LeftEdge
Height = 256 - TopEdge

dove le costanti possono valere rispettivamente anche **320** o **512**, secondo il formato dello schermo in cui è aperta la finestra.

I flag di IDCMP rappresentano un discorso molto lungo, strettamente associato alle tecniche di I/O con Intuition, e ne riparleremo prossimamente. Sinchè si usa la finestra come semplice mezzo di output, non è vitale per ora conoscerne tutti i possibili valori.

Flag di attributo

I flag di attributo della finestra, specificati nel campo Flags tramite costanti sommate con un operatore di somma algebrica (+) o logica (|) sono parecchi e abbastanza complessi. Vediamoli tutti, uno per uno:

SIMPLE_REFRESH. Intuition preserva i contenuti della finestra solo nel caso se ne effettui la trasposizione sullo schermo col mouse.
Se un'altra finestra viene sovrapposta e poi spostata, il contenuto della window sottostante a quella dotata di "restauro semplice" viene perduto.

SMART_REFRESH. Intuition usa sempre la bitmap dello schermo (specificata nella struttura dinamica Window, creata da OpenWindow()) per conservare i dati di visualizzazione, però quando una parte della finestra viene coperta da un'altra finestra o cancellata per un ridimensionamento, Intuition memorizza automaticamente questa porzione "smarrita" in un buffer di memoria creato dinamicamente senza intervento del programmatore.

NOCAREREFRESH. Specificando questo flag, non verranno inviati segnali ad Intuition dal sistema quando la finestra necessita di essere aggiornata.

SUPER_BITMAP. La finestra viene dotata di una bitmap personale da usare invece della normale area dedicata alla visualizzazione, separata.

BORDERLESS. La finestra viene creata priva della cornicetta bianca di delimitazione.

GIMMEZEROZERO. La finestra ha il bordo considerato separato dalla finestra vera e propria, quindi è impossibile disegnare nella porzione di finestra occupata da bordi o gadget.

BACKDROP. La finestra viene collocata in profondità e non può essere traslata, ridimensionata nè modificata in profondità.

ACTIVATE. La finestra viene aperta attivata, come se si cliccase col mouse al suo interno.

RMBTRAP. Ad Intuition viene notificato tramite la porta IDCMP quando l'utente seleziona la finestra e preme il pulsante destro del mouse.

WINDOWSIZING. La finestra viene dotata del gadget di ridimensionamento, che potrà essere svolto nei limiti fissati dai valori inseriti nei campi MinWidth, MaxWidth, MinHeight e MaxHeight.

WINDOWDRAG. La finestra viene dotata del gadget che ne consente la traslazione col mouse.

WINDOWCLOSE. La finestra viene dotata del gadget di chiusura. Per funzionare, questo gadget deve comunque essere associato a del codice del programma che esegua il controllo del traffico di messaggi con la porta IDCMP e usi la funzione CloseWindow() appropriatamente.

WINDOWDEPTH. Inserisce i gadget di cambiamento di profondità.

REPORTMOUSE. Il task che controlla la finestra verrà tenuto al corrente, tramite IDCMP, degli spostamenti del mouse.

Il campo **FirstGadget** della struttura NewWindow punta al **primo** elemento di una lista di gadget definiti dal programmatore, come vedremo più avanti.

Il campo **Checkmark** è raramente usato e consente di specificare una sagoma personalizzata per il simbolo di "spunta" nelle opzioni dei menu associati alla finestra.

Il campo **Title** è un puntatore alla stringa C (chiusa da un byte a NULL) che contiene l'intestazione della finestra. Il puntatore Screen vale NULL quando la finestra deve essere inserita nello schermo del Workbench, altrimenti deve contenere un puntatore valido alla struttura di uno schermo aperto precedentemtene con un OpenScreen().

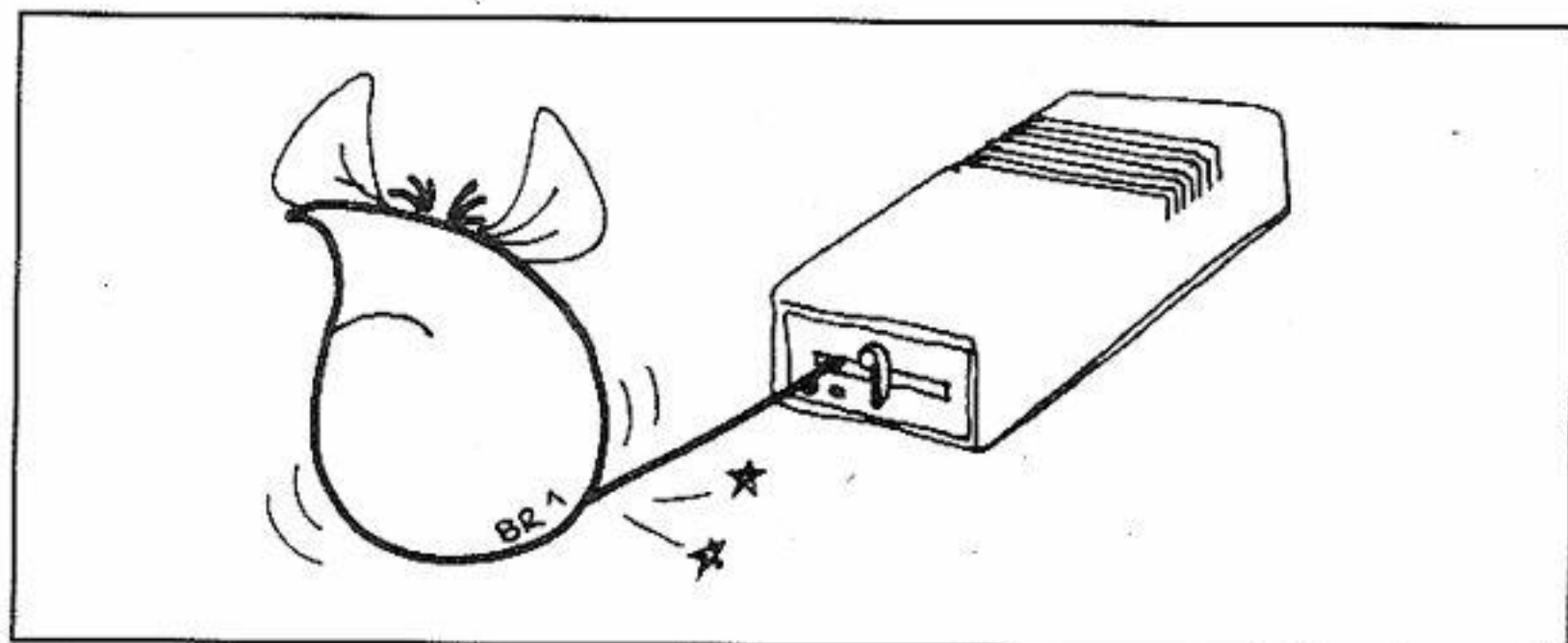
Il puntatore BitMap deve contenere NULL se non si usa il modo SuperBitmap.

Il Type della finestra può essere WBENCHSCREEN, ad indicare che lo schermo va nella finestra del Workbench, oppure CUSTOMSCREEN se deve andare in uno schermo aperto all'uopo.

Apriamo una finestra

Il secondo listato illustra praticamente come si apre una finestra nello schermo del Workbench ed è molto simile a quello relativo alla apertura di uno schermo, data la già citata *simmetria* della filosofia di gestione di screen e window da parte di Intuition.
Analizzandolo, si noterà come sia relativamente semplice definire una finestra, aprirla e chiuderla da programma. Si badi bene che, proseguendo ed ampliando il discorso sulla necessità di *chiudere* solo quello che è stato effettivamente *aperto*, se si tenta di chiudere uno schermo senza prima averne chiuso tutte le eventuali finestre interne, si ottiene una Guru Meditation.

Conclusioni

Per questo mese la lunga chiacchierata introduttiva su Intuition è terminata.
In seguito proseguiremo la trattazione pratica di questa libreria con l'illustrazione dell'uso pratico di varie altre funzioni in listati brevi e semplici e delle caratteristiche di Intuition.

```
/* Apre uno schermo definito dall'utente */
/* ************************************* */
#include <intuition/intuition.h>
#ifdef LATTICE
# include <proto/intuition.h>
# include <proto/exec.h>
#else
# include <functions.h>
#endif
struct IntuitionBase *IntuitionBase;
struct Screen *schermo;
struct NewScreen strutscher = {
  0,0,            /* origine X,Y                */
  640,512,        /* Larghezza ed altezza       */
  3,              /* Profondita' in bitplanes   */
  0,1,            /* DetailPen e BlockPen       */
  LACE+HIRES,     /* Modi visualizzazione       */
  CUSTOMSCREEN,   /* Tipo di schermo            */
  NULL,           /* Fonte di default           */
  "Mioschermo",   /* Intestazione               */
  NULL,           /* Punta primo gadget custom  */
  NULL            /* Punta struttura Bitmap     */
};
```

```
void die( LONG coderet )
{
if (schermo) CloseScreen( schermo );
if (IntuitionBase) CloseLibrary( IntuitionBase );
  _exit( coderet );
}

void _main()
{
  IntuitionBase = ( struct IntuitionBase * )\
       OpenLibrary( "intuition.library", 0L );

  if ( IntuitionBase == NULL )
     die( 10L );

schermo=(struct Screen *)OpenScreen(&strutscher);

  if ( schermo == NULL )
     die( 11L );

  Delay( 50L * 5 );
  die( 0L );
}
```

Apertura di uno schermo
Ecco il breve listato, ovviamente in "C", per aprire uno schermo su Amiga. Si noti l'uso intensivo delle numerose istruzioni descritte nell'articolo

```
/* Programma apertura finestra custom */
/* ********************************** */
#include <intuition/intuition.h>
#ifdef LATTICE
# include <proto/intuition.h>
# include <proto/exec.h>
#else
# include <functions.h>
#endif
struct IntuitionBase *IntuitionBase;
struct Window *finestra;
struct NewWindow strutfin = {
  101,36,          /* Origine XY              */
  409,124,         /* Larghezza e Altezza     */
  0,1,             /* DetailPen e BlockPen    */
  NULL,            /* Flags di IDCMP          */
  WINDOWSIZING+WINDOWDRAG+WINDOWDEPTH+\
  WINDOWCLOSE+ACTIVATE+NOCAREREFRESH,
  NULL,            /* Primo gadget            */
  NULL,            /* Checkmark personale     */
  "Miafinestra",   /* Intestazione            */
  NULL,            /* Schermo custom          */
  NULL,            /* Bitmap custom           */
  0,0,             /* Min largh. e altez.     */
  0,0,             /* Max largh. e altez.     */
  WBENCHSCREEN     /* Tipo schermo destin. */
};
void die( LONG t )
{
  if (finestra) CloseWindow(finestra);
  if (IntuitionBase) CloseLibrary(IntuitionBase);
  _exit( t );
}
void _main()
{
  IntuitionBase = (struct IntuitionBase *)\
     OpenLibrary( "intuition.library", 0L );

  if ( IntuitionBase == NULL )
     die( 10L );

 finestra=(struct Window *)OpenWindow(&strutfin);
 if ( ! finestra )
     die( 11L );

 Delay( 5  * 50L );
 die( 0L );
}
```

Apertura di una finestra in Amiga "C"
Ed ecco invece, in un listato altrettanto breve, la possibilità di aprire una finestra. Si noti la rassomiglianza concettuale tra i due programmi "C".

Come si stampa oggi, e perché

< di Alessandro de Simone >

< Esaminare C. C. n. 84 "Alla ricerca del linguaggio perduto" >

Chi avrebbe mai pensato che la realizzazione di un giornale, pur se di aspetto "domestico", sarebbe stata possibile addirittura in casa nostra?

E' passato molto tempo da quando, con il C/64 ed un drive 1541 (oltre ad uno dei tanti programmi grafici del tempo andato, come **Print Master**, **Print Shop**, **News Room**, lo stesso **Geos**) alcuni fogli di carta uscivano asmaticamente dalla storica stampante MPS-803 nel disperato tentativo di simulare la stampa di un giornale.

A quei tempi, infatti, le ore indispensabili per il parto di una pagina pseudo grafica facevano gridare al miracolo e, contemporaneamente, ponevano in secondo piano il pensiero di poter realizzare le stesse cose limitandosi a lavorare con carta, trasferibili autoadesivi ed un po' di pazienza.

La notevole velocità delle moderne stampanti, il loro relativo basso prezzo, la diffusione di numerosi software specifici, che consentono la simulazione di veri linguaggi descrittori di pagina (vedi l'articolo "*Alla ricerca del linguaggio perduto*", una chiacchierata sul **Postscript** pubblicata sul **N. 84**) rendono reale, di fatto, la possibilità di realizzare a casa propria una qualsiasi pubblicazione, anche se limitatamente al formato A 4 (cm. 21 x 29.6) e con una qualità più che dignitosa.

Non solo Postscript

Non tutti sanno, infatti, che la precisione grafica conseguibile con una stampante laser può essere **simulata**, pur se con gli ovvi limiti di qualità del caso, con una **qualsiasi 9 aghi**, anche di prezzo più che basso.

Sia per i computer **Ms - Dos**, sia per **Amiga**, sono oggi disponibili programmi in grado di interpretare la pagina di testo (o di grafica) realizzata in linguaggio Postscript (o altri cosiddetti "descrittori di pagina") e stamparla su una qualsiasi stampante.

Rimandando il lettore ad altri articoli pubblicati in precedenza (e ad altri sparsi qua e là in questo stesso numero), ci teniamo a sottolineare che, per ottenere una pagina Postscript, **non** è affatto necessario imparare il linguaggio stesso (ci mancherebbe altro!), ma è sufficiente cliccare in qualche finestra ed attivare alcune voci di qualche menu.

Tutti i programmi professionali che si rispettano, infatti, tra le varie stampanti di cui consentono la selezione, offrono la possibilità di indicare la "voce" Postscript.

Supponiamo, ad esempio, di operare con un programma di grafica (parliamo, ovviamente, di software di un certo pregio, e cioè quasi tutti quelli "seri"...) e di aver indicato, come stampante, un modello Postscript qualunque, anche se, in effetti, la stampante collegata al nostro computer è una modestissima 9/24 aghi.

Al momento di stampare il vostro lavoro, il programma offre, di solito, la possibilità di inviare subito, sulla stampante in quel momento attiva, il disegno eseguito; oppure di riversare, su disco, il file stesso in formato Postscript.

Se, a questo punto, selezionate la stampante come output, vedrete apparire sul foglio di carta posizionato nella stampante ad aghi (a meno di un crash del sistema...) numerosi comandi, apparentemente privi di significato.

Si tratta, a leggerli bene, di **istruzioni Postscript**: il computer, infatti, ritiene di inviare un codice leggibile ad una stampante che, invece, non è in grado di interpretare la sequenza di comandi ASCII e li stampa come se fosse un qualsiasi file di testo.

E' come se, insomma, venisse stampato **il programma** e non il suo **"effetto"**!

Vari esempi di crenatura (modifica di spazi tra i caratteri)

AVAV

AVAV

AVAV

AVAV

AVAV

Una stampante Postscript, invece, è solo in grado di interpretare un file ASCII che rappresenti un vero e proprio programma; se, al contrario, inviate un comune file di testo ASCII (come, ad esempio, una lettera scritta con un qualsiasi Text Editor) ad una stampante Postscript, questa andrà in tilt perché tenterà disperatamente di interpretare il contenuto del file!

Dunque: un programma in grado di manipolare istruzioni Postscript consente, di solito, non solo l'invio verso la stampante di una pagina da stampare... (come è ovvio che sia) ma consente, anche, di registrare il file ASCII corrispondente alle istruzioni Postscript necessarie per stampare la pagina stessa.

In seguito, con un banale `Print NomeFile` (o altri comandi similari, disponibili in qualsiasi Sistema Operativo come **Ms - Dos**, **AmigaDos**, eccetera) è possibile inviare il file stesso senza usare nessun programma specifico; è come se utilizzassimo il comando `Type`, insomma, per visualizzare il contenuto di un file di testo: nel nostro caso, però, viene stampata una pagina Postscript, con tutti i vantaggi che è possibile immaginare. Chi non possiede una stampante Postscript, come già anticipato, può ugualmente vedere apparire sulla sua modesta 9/24 aghi una pagina "descritta" con tale linguaggio, a patto di usare una delle

Questo che vedete è denominato, a seconda dei libri o delle riviste che avrete occasione di leggere, **Riquadro**, oppure **Frame** o **Box**. In tutti i casi, comunque, si tratta di un **rettangolo** in cui viene inserito un testo. Di solito il testo del riquadro, per esigenze estetiche, viene allineato con il testo "normale", cioè quello che potete vedere nelle colonne di questa stessa pagina. Si osservi, infatti, la linea inserita di proposito per dimostrare il corretto allineamento del testo di questo box con quello "normale" del resto della pagina.

Questo, invece, è un frame contenente un testo che non è allineato con il testo della pagina (si noti la linea); si tratta di finezze, ma "fare" del DTP significa anche questo...

Osservate il testo della pagina presente al di sopra di questo frame: non è allineato (vedi linea) a causa della presenza di altri frame, responsabili dello "sfasamento" del testo.

Crenatura (in inglese: Kerning)
*Si noti come, modificando la spaziatura tra i caratteri, è possibile avvicinare tra loro quelle lettere che, per la loro particolare forma, consentono parziali sovrapposizioni. Le linee verticali visibili tra **A** e **V** permettono di meglio evidenziare il Kerning.*

Caratteristiche principali di un carattere

In questa figura sono indicati i principali elementi che caratterizzano un **Tag** (tradotto spesso, in italiano, con il termine "**Marcatore**"). Vi sono diverse dimensioni da considerare in tipografia.

L'**altezza** del **corpo** va misurata dal punto più alto dei caratteri dotati di ascendenti (come: b, d, f, h, l, t) a quello più basso dei caratteri dotati di discendenti (come: g, p, q, y, j). L'altezza delle **Maiuscole**, invece, è minore della precedente a causa della mancanza di discendenti nelle maiuscole stesse; per ciò che riguarda la crenatura (Kerning) vedere gli altri riquadri.

Vedove e orfani (a)

Per **"Paragrafo"**, nel linguaggio dell'editoria, si intende comunemente quella parte di testo compresa tra due "a capo".

Questo che vedete in sottolineato, ad esempio, è un paragrafo perché inizia subito dopo un "a capo" e termina con un altro (punto) "a capo".

Di solito un paragrafo, per comodità di lettura, deve essere riportato per intero in una colonna.

Se ciò non è possibile, ovviamente, il paragrafo viene spezzato in due o più parti.

Se, però, il "passaggio" alla colonna successiva è tale da lasciare una sola riga in quella precedente, l'intero paragrafo viene trasportato alla colonna successiva, lasciando un "vuoto" in quella precedente.

Allo stesso modo, se nella nuova colonna dovesse comparire, in seguito all'interruzione, una sola riga, l'intero paragrafo viene trasferito in "cima" alla nuova colonna.

Ciò è evidente nei due frame che vedete uno sotto l'altro (contenenti testo identico, su quattro colonne), in cui l'allineamento viene artificiosamente realizzato unendo o separando paragrafi successivi.

Se ciò non fosse stato fatto (come, appunto, nel box superiore) l'allineamento non sarebbe stato garantito a causa, appunto, dell'impostazione di vedove e orfani.

Il numero di righe, di solito, può essere impostato con una certa libertà.

Vedove e orfani (b)

Per **"Paragrafo"**, nel linguaggio dell'editoria, si intende comunemente quella parte di testo compresa tra due "a capo".

Questo che vedete in sottolineato, ad esempio, è un paragrafo perché inizia subito dopo un "a capo" e termina con un altro (punto) "a capo".

Di solito un paragrafo, per comodità di lettura, deve essere riportato per intero in una colonna.

Se ciò non è possibile, ovviamente, il paragrafo viene spezzato in due o più parti.

Se, però, il "passaggio" alla colonna successiva è tale da lasciare una sola riga in quella precedente, l'intero paragrafo viene trasportato alla colonna successiva, lasciando un "vuoto" in quella precedente. Allo stesso modo, se nella nuova colonna dovesse comparire, in seguito all'interruzione, una sola riga, l'intero paragrafo viene trasferito in "cima" alla nuova colonna.

Ciò è evidente nei due frame che vedete uno sotto l'altro (contenenti testo identico, su quattro colonne), in cui l'allineamento viene artificiosamente realizzato unendo o separando paragrafi successivi.

Se ciò non fosse stato fatto (come, appunto, nel box superiore) l'allineamento non sarebbe stato garantito a causa, appunto, dell'impostazione di vedove e orfani.

Il numero di righe, di solito, può essere impostato con una certa libertà.

utility disponibili in commercio (o nel circuito del Pubblico Dominio, PD) come il famoso **Freedom Of Press** per i sistemi Ms - Dos.

In questi casi, il programma simulatore non fa altro che chiedere il modello di stampante realmente disponibile ed il nome del file che rappresenta la pagina in formato Postscript; dopo un tempo proporzionale alla complessità della pagina stessa, alla velocità del computer ed alla Ram disponibile, sarà possibile ammirare la pagina in tutto il suo splendore.

Non illudetevi, però, di attendere un tempo medio minore di **cinque minuti** per ogni pagina stampata: a qualcosa bisogna pur rinunciare...

Conoscere il carattere

Chi inizia ad operare con un programma impegnativo, come un **CAD** (**C**omputer **A**ided **D**esign = disegno supportato da computer) oppure un **DTP** (**D**esk **T**op **P**ublishing = impaginazione complessa di documenti), spesso trascura numerosi particolari che, al contrario, sono indispensabili per l'armonica composizione di una pagina di testo e di grafica.

In queste pagine, pertanto, riportiamo, nel modo più semplice possibile, una casistica di numerose possibilità offerte dai moderni pacchetti di impaginazione professionale, tutte racchiuse in pratici frame (indicati prevalentemente con i termini Struttura, Box ed altri ancora).

Riportiamo, ovviamente, anche la terminologia usata negli ambienti tipografici ed indichiamo alcuni suggerimenti per realizzare facilmente, ed efficacemente, una pagina in DTP.

Inutile sottolineare che queste note saranno utili per tutti i lettori, indipendentemente dal computer utilizzato o dal particolare pacchetto professionale usato: operando in ambiente Postscript (o simulato che sia) l'universalità dei risultati è indiscutibile.

Unità di misura

L'unità di misura standard per le **fonti** (cioè le particolari "forme" dei vari set di caratteri disponibili) è il **corpo**, corrispondente ad **1/72** di pollice (quest'ultima misura **2.54** centimetri). Il corpo si misura dal punto più alto del più alto ascendente appartenente al set di caratteri riproducibile, al punto più basso del più basso discendente.

Un qualsiasi carattere di **corpo 9**, come il testo che state ora leggendo, è quindi inscrivibile in una "matrice" alta...

$$Corpo = 9 \cdot 2.54 \cdot \frac{1}{72} = 0.3175\ cm$$

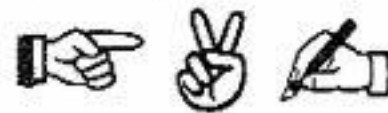
Se, infatti, vi armate di doppio decimetro (e di lente di ingrandimento...) potete constatare quanto affermato.

Per aiutarvi, verificate la riga che segue...

ltdfhk ABCDE gqypj weruioaszxcvnm

...formata, appunto, di caratteri dotati di ascendente, lettere maiuscole, caratteri dotati di discendente e, infine, caratteri privi di ascendente e discendente.

In questa prima chiacchierata sugli elementi di tipografia (che non sono, ovviamente, tutti, ma solo i principali) abbiamo affrontato i temi più interessanti. Vi consigliamo di valutare gli output dei vostri Word Processor preferiti e, perché no?, quelli generati da programmi di DTP o semi grafici in vostro possesso.

Ne vedrete delle belle!

Interlinea

L'interlinea è la distanza tra due punti omologhi di due righe di testo posizionate una sotto l'altra. Di solito l'interlinea vale **1.2 / 1.1** volte l'altezza del corpo cui si riferisce.

Ad esempio, per un corpo **9** l'interlinea è di **1.2 x 9 =10.8** punti. L'importante, insomma, è che il discendente di un carattere presente nel rigo superiore non tocchi l'eventuale ascendente di un carattere posto in corrispondenza del rigo inferiore. Nella "figura" qui in basso, le due linee distano tra loro circa 6 millimetri misura che, per un corpo 15, è quella giusta:

(15 x 2.54 / 72) x 1.1 = 5.8208...

Per corpi relativamente grandi, infatti, il coefficiente vale **1.1** (10%).

Corpo 15 Corpo 15
Corpo 15 Corpo 15
Corpo 15 Corpo 15

Differenti dimensioni tra varie altezze di uno stesso Font

Helvetica a b c d e f g h i l m n o p q A B C D E F G H I L M N O P Q (corpo 9)
Helvetica a b c d e f g h i l m A B C D E F G H I L M **(corpo 12)**
Helvetica a b c d e f g A B C D E F G **(corpo 15)**
Helvet. a b c d e f A B C D E F **(corpo 18)**
Hel. a b c d e f A B C D E F **(corpo 20)**
Helve. a z A Z **(corpo 30)**

Alcuni Font sono disponibili solo in corsivo, come ad esempio il Font...

ITC Zapf Chancery Medium Italic
A B C D E F G H a b c d e f g h j

I font più noti

In basso alcuni esempi (tutti in corpo 9) dei più noti font disponibili sulla maggior parte delle stampanti laser. Pur se esistono centinaia di font diversi, quelli riportati sono più che sufficienti per impostare correttamente qualsiasi pubblicazione. **Computer Club**, ad esempio, ne usa solo cinque!

Carattere Times Roman
Carattere Times Roman neretto
Carattere Times Roman corsivo
Caratt. Times Roman ner. cors.

Helvetica narrow
Helvetica narrow neretto
Helvetica narrow corsivo
Helvetica narrow ner. cors.

Helvetica
Helvetica neretto
Helvetica corsivo
Helvetica ner. cors.

ITC Bookman Demi
ITC Bookman Demi neretto
ITC Bookman Demi corsivo
ITC Bookman Demi ner. cors.

Carattere Symbol
αβχδεφγηιλμνοπθρστυϖζξψφκω

Carattere Courier
Carattere Courier neretto
Carattere Courier corsivo
Caratere Cour. ner. cors.

Palatino Roman
Palatino Roman neretto
Palatino Roman corsivo
Palatino Roman ner. cors.

Carattere ITC Zapf Dingbats

❁✪✻✼❄❅❆❇●❍■❏❐❑❒▲▼
◆❖▌❈❉❊✿✢✣✤✥✦✧★✩✪✫✬✭

New Century Schoolbook Roman
New Cent. School. Roman ner.
New Cent. School. Roman corsivo
New Cent. Sc.. Rom. ner. cors.

✔ Amiga	❑ Principianti	❑ Esperti	✔ Tutti	✔ HELP
✔ Ms - Dos				
❑ Recensioni				
✔ Hardware				
❑ Software				
❑ Applicazioni				
✔ Programmazione				
❑ Dos				
❑ Pascal				
❑ C				
❑ Basic				
❑ Assembly				

La periferica più trascurata dall'utente "medio" nasconde spesso potenzialità enormi. Vediamo di scoprirle un po' per volta.

La stampante, iniziamo a conoscerla

< di Ascanio Orlandini >

< Inizia una nuova rubrica di utilità generale >

Oggigiorno la stampante è una delle periferiche più diffuse tra gli utenti informatici: chi non ne è già in possesso, certamente l'avrà inserita nella lista dei prossimi acquisti.

Ecco, quindi, una serie di articoli destinati a conoscere la propria stampante in modo più approfondito, senza limitarsi ad usarla tramite altri programmi (operazione che già svolge quotidianamente), ma per realizzare programmi che siano in grado di pilotarla direttamente a basso livello.

Cercheremo di togliere quel velo di misterioso fascino che riveste la stampante, una periferica per troppo tempo data per scontata; tutti gli utenti certamente sapranno usare chissà quale elaboratissimo Word Pocessor per riportare su carta una lettera, o qualsiasi altro testo, ma ben pochi sono in grado di impiegare concretamente, da programma, tutte qualle avanzatissime funzioni che le stampanti odierne mettono a disposizione del programmatore.

In questo primo articolo non solo saremo brevi e concisi, ma resteremo un po' sul generico ripromettendoci, nei prossimi fascicoli, di proporre esempi dimostrativi concreti.

Nel corso degli esempi faremo riferimento a stampanti molto diffuse (come la **Star NB24-10** a 24 aghi) e ad altre ragionevolmente conosciute, tali da possedere caratteristiche comuni a tantissime altre stampanti, anche di alta qualità.

Consigliamo di tenere comunque sottomano, nella lettura degli articoli, il manuale della vostra stampante per verificare, di volta in volta, le analogie e le differenze con quella presa di esempio, qualora fosse diversa o richiedesse parametri differenti.

Tipi di stampanti

Caratteristica essenziale per il buon funzionamento della stampante è che computer e periferica parlino la *stessa lingua*, ossia che entrambi siano configurati correttamente.

Per compiere questa operazione di settaggio, o di **configurazione** che dir si voglia, bisogna agire tanto sul computer quanto sulla stampante.

Una stampante, generalmente, mette a disposizione differenti **emulazioni**; ciò significa che può comprendere il linguaggio di diversi tipi di stampanti anche se di marche differenti.

Ad esempio, quasi tutte le stampanti emulano la **Epson LQ-1000** (modo standard), la **IBM ProPrinter** e la **IBM GraphicPrinter**. La selezione del tipo di emulazione di cui intendiamo servirci viene effettuata mediante il posizionamento di due **dip-switch**, speciali mini-interruttori di solito affiancati in gruppi tra loro, che consentono di imporre la configurazione di base che la stampante deve assumere.

Gli switch vengono posizionati, dal costruttore, in luoghi spesso non facilmente individuabili.

Per accedervi (ci riferiamo alla Star LC24-10) è necessario estrarre la cartuccia del nastro inchiostrato e sollevare una lamina di plastica.Chi possiede un modello diverso (o di altra marca) può trovare informazioni utili sulla loro ubicazione riferendosi all'indice analitico del manuale della stampante, sotto la voce, appunto, dip-switch.Molte stampanti offrono la comoda opportunità di accedere ai dip - switch attraverso piccoli sportelli praticati sulla parte esterna della periferica.

In ogni caso, però, è sempre necessario far riferimento al manuale dal momento che la successione ordinata di una decina di interruttori non viene (quasi) mai accompagnata da un'etichetta che ne spiega la funzione.

Ricordiamo, a chi volesse effettuare esperimenti per proprio conto, di segnare chiaramente, su un foglio di carta, la corretta disposizione degli interruttori (acceso / spento) per riportarli nella posizione originaria una volta compiute le prove desiderate.

Tale configurazione, che si riferisce all'uso "normale" della stampante, consigliamo anzi di trascriverla anche sul manuale, in modo da averla a portata di mano in qualsiasi momento.

Oltre all'emulazione, caratteristica di rilevanza principale, si possono settare molteplici altri parametri tra cui:

➪ la possibilità di far stampare il carattere zero con la barra trasversale (ø) oppure no (**0**);
➪ l'entità dell'interlinea;
➪ la stampa normale o compressa;
➪ l'impostazione del margine inferiore;
➪ la scelta del set di caratteri;
➪ l'adozione di particolari caratteri nazionali (italiani, USA, tedeschi, eccetera);
➪ l'abilitazione, o meno, della definizione di "soft-font" da parte dell'utente;
➪ l'abilitazione o disabilitazione del controllo della presenza della carta, eccetera.

Le caratteristiche citate sono (quasi) tutte disponibili su ogni modello di qualsiasi marca, pur se dipendono strettamente, per ciò che riguarda la particolare configurazione On / Off degli switch, dalla stampante in vostro possesso. Se fate riferimento al manuale, scoprirete una miriade di possibilità di cui non immaginavate nemmeno l'esistenza.

Amiga + MS - DOS

Mentre su computer MS - DOS, per i nostri scopi, ci si limita ad eseguire la corretta configurazione della stampante nei confronti del calcolatore (fanno eccezione i vari pacchetti applicativi i quali hanno ognuno il proprio metodo per adottare le diverse stampanti), nei calcolatori della serie **Amiga** si deve installare correttamente il così detto **driver di stampa**, che consente all'elaboratore di selezionare le diverse stampanti e che consiste in un vero e proprio **interprete** in grado di tradurre il formato del computer in quello della stampante ad esso collegata.

Tanto per fare un esempio chiarificatore, il codice CLS (CLearscreen, cancella lo schermo) su Amiga è rappresentato da **Control + L** che, attraverso il **driver Epson**, viene trasmesso alla stampante come **Chr$(12)** cioè il codice di **FF** (FormFeed, pagina successiva).

Per installare correttamente il driver, dovete innanzitutto individuare il nome del driver che soddisfa la vostra stampante, facendo ricorso al manuale di Amiga.

Dovrete poi assicurarvi che il driver sia effettivamente presente nella directory **:devs/printers/** del disco che usate abitualmente per lanciare il sistema quando usate la stampante.

Se lì non compare, dovrete cercarlo nella stessa directory dei dischi del **WorkBench** o dell'**Extras** originali e copiarlo sul vostro disco di sistema, sempre nella directory :devs/printers/.

Nella confezione di alcune stampanti sono a volte inseriti dischetti contenenti i driver da utilizzare con i vari computer.

Di solito, però, il dischetto non è presente perché la stampante stessa, potendo quasi sempre emulare uno standard diffuso, funziona egregiamente con altri driver sicuramente in possesso dell'utente.

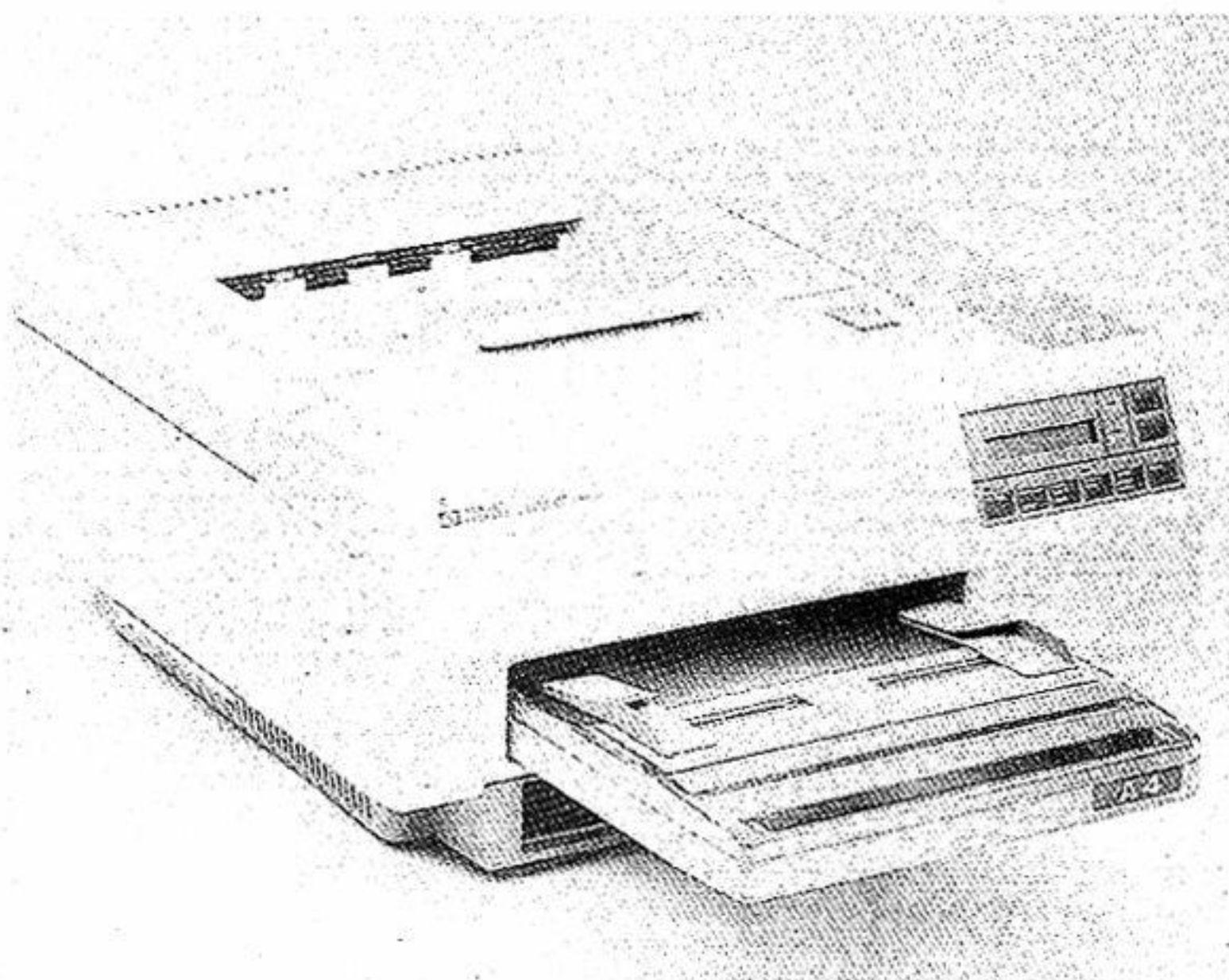
E' il caso, appunto, dei driver contenuti nel Workbench, in grado di pilotare qualsiasi stampante oggi commercializzata.

Eseguita questa operazione preliminare, lanciando il sistema dal disco su cui volete effettuare l'installazione, bisogna lanciare il programma **Preferences**, selezionare il gadget **Change Printer** e agire sulle frecce verticali finchè il driver corretto venga evidenziato. A questo punto basta premere su **OK** e poi su **Save**.

Se, tra quelli elencati, non individuate il driver che occorre significa che non è fisicamente presente nella directory devs:printers/.

E' bene precisare ancora una volta che parlando di *devs:* ci riferiamo alla directory **devs** del disco da cui il sistema si è avviato dopo un reset, oppure all'accensione. **Se, cioè, cambiate dischetto, la stampante potrebbe non funzionare correttamente.**

Nel momento in cui attivate Preferences, questo cerca i driver nella directory printers contenuta, appunto, in devs:; dovrete quindi assicurarvi di avere avviato il sistema proprio dal disco su cui volete apportare l'installazione.

Il discorso può apparire, ad un novizio di Amiga, più contorto di quanto sia concretamente. Abituatevi, comunque, a questo modo di ragionare, in quanto fa parte del modo di operare del meraviglioso computer. ✍

I risultati dell'inchiesta

Che cosa hanno detto i nostri lettori nel formulario pubblicato nei numeri 85 e 86 di Computer Club

Mentre scriviamo queste note continuano ad arrivare le vostre schede che hanno ormai raggiunto una quantità ragguardevole e, come tale, suscettibile di attendibili deduzioni.

Anzitutto, coloro che desiderano restare ancorati al **C/64** sono molto meno di quanto credevamo all'inizio dell'anno. Un misero (nel senso statistico, senza offesa per nessuno...) **4.61** per cento rende ardua la nostra decisione di insistere nel parlare dell'obsoleto computer fino alla fine di quest'anno. Ma ogni promessa è debito, e va rispettata. Ci pensino, semmai, i più testardi, e prendano le decisioni del caso.

I possessori di solo **Amiga** rappresentano una maggioranza consistente (**61.54%**) rispetto agli utenti di solo **MS - DOS (21.54%)**. La sorpresa maggiore, invece, è costituita dal **12.31%** di entusiasti utilizzatori di *entrambi* i sistemi. In pratica, dei nostri lettori, il **73.85%** usa e conosce **Amiga**, il **33.85** usa e conosce MS - DOS.

Il dato verrà esaminato più in dettaglio nel corso delle prossime inchieste che, come notate, sono diventate un vero e proprio "spazio fisso".

All'interno della fascia Amiga, il **10%** possiede un **hard disk**; solo il 5% è rimasto fermo alla configurazione base (drive interno, 500 Kram), un altro **5%** ha preferito acquistare un secondo drive lasciando inalterata la quantità di memoria RAM. Tutti gli altri Amighi dispongono di almeno 1 mega e di due drive per floppy da 3.5 pollici in totale. Oltre il **30%** degli utenti Amiga dispone di più di 2 mega di Ram. Si noti che la somma delle percentuali citate **non** è 100 perché le categorie, per come erano poste le domande, risultano parzialmente "sovrapposte" tra loro.

Per quanto riguarda la **stampante** utilizzata, il **54%** dei lettori ha dichiarato di essersi equipaggiata con stampanti a 24 aghi (o migliori). A causa della imprecisione della domanda sul possesso, o meno, della stampante a **9 aghi** (mea culpa, mea maxima culpa...) non è possibile stabilire quanti lettori hanno una stampante a 9 aghi oppure non la posseggono affatto. Provvediamo all'errore, come notate, con l'inchiesta di questo mese.

Per ciò che riguarda l'utenza MS - DOS, *quasi tutti* posseggono l'hard disk; *quasi nessuno* si è limitato alla configurazione base di RAM (640 K / 1 mega) ma ci si è orientati dai 2 mega in su.

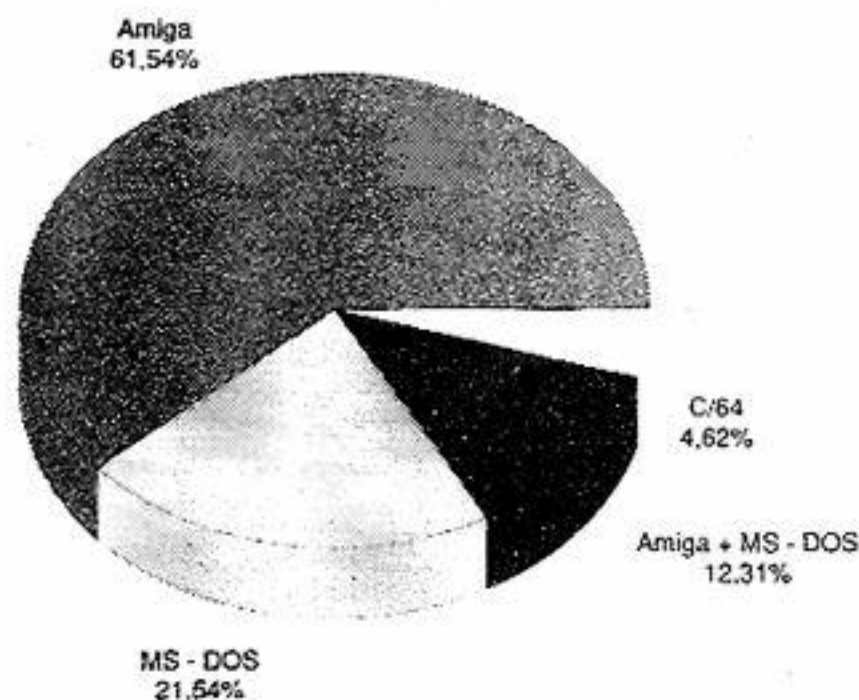
Per ciò che riguarda gli **argomenti preferiti**, la risposta si è ugualmente suddivisa tra *didattica* e *utilizzo ottimale dei pacchetti più diffusi*.

In minoranza (**18.5%**) le richieste di recensioni di hardware e di software, percentuale in contrasto con la richiesta di aumentare lo spazio a favore dell'utilizzo ottimale: le recensioni non sono, in fin dei conti, dei mini-manuali d'uso? *Nessuno*, o quasi, richiede videogame: grazie, ragazzi!

Per ciò che riguarda il **prezzo di copertina** relativo all'inclusione di un dischetto, beh, forse non siamo stati capiti da una (pur sostanziosa) minoranza: come si fa a chiedere che rivista + dischetto venga posta in vendita al prezzo di sole lire 6000? Ma anche qui è probabile che vi sia una difficoltà di interpretazione della domanda; saremo sempre più chiari in futuro.

La gran maggioranza degli intervistati, invece, è favorevole ad un dischetto di supporto ed il prezzo suggerito, in certi casi, è addirittura superiore a quello che ci saremmo aspettati.

Quasi il **34%** dei lettori, infine, dichiara di possedere un **modem** o di prevederne l'acquisto nel prossimo 1992.

Ringraziamo, quindi, tutti coloro che hanno partecipato all'inchiesta e li sollecitiamo a continuare nel rispondere con la massima sincerità a tutte le inchieste che seguiranno nei prossimi numeri: i risultati verranno pubblicati a mano a mano che giungeranno, in Redazione, le informazioni necessarie.

La stampante, usi e costumi

Inchiesta Settembre'91

Stavolta vi chiediamo di rispondere alla nostra inchiesta sulle stampanti. Ringraziamo i numerosissimi lettori che hanno aderito alle due precedenti inchieste incoraggiandoci a continuare sulla strada intrapresa. Alcuni di loro, come riportato in altra parte di questo stesso questionario, sono stati "premiati". Non abbiate timore a staccare il foglio: seguendo la linea tratteggiata eviterete che la pagina "opposta" si distacchi dal fascicolo. Chi lo desidera, ovviamente, potrà inviare la fotocopia riproducente l'inchiesta.

Domanda N. 1 (età)

Quale è la tua età?

❏ meno di 18 anni.

❏ tra i 18 ed i 24 anni.

❏ oltre i 24 anni.

Domanda N. 2 (attività)

Quale attività svolgi?

❏ Studente (o in attesa di occupazione).

❏ Impiegato / operaio/ professionista / tecnico.

❏ Pensionato.

Dom. N. 3 (computer che USERAI)

Quale computer pensi di usare nel corso del 1992?

❏ C/64, C/128, Spectrum o altri a 8 bit.

❏ **Amiga** ❏ Amiga + scheda Ms - Dos

❏ **Ms - Dos** 8088/86 ❏ 80286 ❏ 80386/486

Lettori... fortunati

Nonostante non avessimo promesso alcunchè, abbiamo deciso egualmente di **"premiare"** alcuni lettori (presi a caso tra i nominativi delle schede pervenute) che, rispondendo alla nostra richiesta, hanno inviato il questionario debitamente compilato. Riceveranno diverse pubblicazioni della Systems Editoriale, pertanto: **Davide Villa** (Concorezzo); **Cosimo Napolitano** (Brindisi); **Gian Paolo Jesi** (Ferrara); **Marcel Alfred Denrath** (Cagliari); **Gianni Sarto** (Lombardore - To).

Dati del lettore (compilazione facoltativa)

Cognome

Nome

Indirizzo

(CAP) Città **Tel.**

Domanda N. 4 (stampante usata OGGI)

Quale stampante usi abitualmente?

- ❑ Un modello a 9 aghi
- ❑ Nessuna
- ❑ Un modello a 24 aghi (o comunque di medio prezzo)
- ❑ Una laser (o un modello di prezzo elevato)

Dom. N. 5 (stam. che userai nel 1992)

Quale stamp. userai nel corso del prossimo 1992?

- ❑ Un modello a 9 aghi
- ❑ Nessuna
- ❑ Un modello a 24 aghi (o comunque di medio prezzo)
- ❑ Una laser (o un modello di prezzo elevato)

Domanda N. 6 (uso della stampante)

Quale uso fai abitualmente della stampante?

- ❑ Non ho la stampante
- ❑ Stampa listati (gener.)
- ❑ Grafica
- ❑ Desk Top Publishing
- ❑ Word Processing
- ❑ Data Base, Spreadsheet

Domanda N. 7 (uso ideale)

Quale uso, invece, vorresti farne?

- ❑ Stampa listati
- ❑ Stampa tabulati (gener.)
- ❑ Grafica
- ❑ Desk Top Publishing
- ❑ Word Processing
- ❑ Data Base, Spreadsheet

Dom. N. 8 (programmi professionali)

Quale programma professionale utilizzi abitualmente? (indicare un solo nome per ciascuna categoria)

- ❑ Word Processor:
- ❑ SpreadSheet:
- ❑ Data Base:
- ❑ Desk Top Publishing:
- ❑ Grafica:
- ❑ Altro software (specificare tipo e nome):

Dom. N. 9 (argomenti preferiti)

A quali argomenti (in cui la stampante è "protagonista") vorresti che dedicassimo più spazio?

- ❑ Ritengo che lo spazio dedicato sia sufficiente.
- ❑ Recensioni di nuovi modelli di stampante.
- ❑ Didattica (uso della stampante nei vari linguaggi: C, Basic, Pascal, Assembly, ecc.).
- ❑ Uso ottimale di programmi ed utility commercializzati (Word Processor, DTP, Spreadsheet, Database, ecc.).

(E' possibile barrare più risposte, purchè non in contraddizione tra loro).

Inserire in busta chiusa, affrancare e spedire a:

Systems Editoriale
Via Mosè 22
cap 20090 Opera (Mi)

Ritieni che gli articoli che pubblichiamo sulle novità, recensioni, impressioni d'uso, siano di aiuto per la scelta dei prodotti che intendi acquistare?

- ❑ Sì
- ❑ Seguo consigli di amici
- ❑ No
- ❑ Seguo consigli del negoziante

SYSTEMS EDITORIALE PER TE

La voce
Aggiunge al C/64 nuovi comandi Basic che consentono sia di far parlare il computer, sia di farlo Cantare! Diversi esempi allegati.

Cassetta: L. 12000 - Disco: L. 15000

Raffaelio
Un programma completo per disegnare, a colori, con il C/64: linee, cerchi, quadrati, eccetera. Valido sia per disegno a mano libera che geometrico.

Cassetta: L. 10000

Oroscopo
Devi solo digitare la data di nascita e le coordinate geografiche del luogo che ti ha dato i natali. Vengono quindi elaborate le varie informazioni (case, influenze dei segni astrali, eccetera) e visualizzato un profilo del tuo carattere. Valido per qualsiasi anno, è indicato sia agli esperti sia ai meno introdotti. E' allegata una tabella delle coordinate delle più note città italiane e l'elenco delle ore legali in Italia dal 1916 al 1978.

Cassetta: L. 12000 - Disco: L. 12000

Computer Music
Cassetta contenente numerosi brani di successo da far eseguire, in interrupt, al tuo C/64 sfruttando, fino in fondo, il suo generatore sonoro (SID).

Cassetta: L. 12000

Gestione Familiare
Il più noto ed economico programma per controllare le spese e i guadagni di una famiglia.

Cassetta: L. 10000 - Disco: L. 10000

Banca Dati
Il più noto ed economico programma per gestire dati di qualsiasi natura.

Cassetta: L. 10000 - Disco: L. 10000

Matematica finanziaria
Un programma completo per la soluzione dei più frequenti problemi del settore.

Cassetta: L. 10000 - Disco: L. 20000

Analisi di bilancio
Uno strumento efficace per determinare con precisione i calcoli necessari ad un corretto bilancio.

Cassetta: L. 10000 - Disco: L. 20000

Corso di Basic
Confezione contenente quattro cassette per imparare velocemente le caratteristiche delle istruzioni Basic del C/64 e i rudimenti di programmazione. Interattivo.

Cassetta: L. 19000

Corso di Assembler
Un corso completo su cassetta per chi ha deciso di abbandonare il Basic del C/64 per addentrarsi nello studio delle potenzialità del microprocessore 6502. Interattivo.

Cassetta: L. 10000

Logo Systems
Il linguaggio più facile ed intuitivo esistente nel campo dell'informatica; ideale per far avvicinare i bambini al calcolatore.
Diversi esempi allegati.

Cassetta: L. 6500

Compilatore Grafico Matematico
Uno straordinario programma compilatore, di uso semplicissimo, che permette di tracciare, sul C/64, grafici matematici Hi-Res ad altissima velocità. Esempi d'uso allegati.

Cassetta: L. 8000

Emulatore Ms-Dos e Gw-Basic
Un prodotto, unico nel suo genere, che permette di usare, sul C/64 dotato di drive, la sintassi tipica del più diffuso sistema operativo del mondo. Ideale per studenti.

Solo su disco: L. 20000

Emulatore Turbo Pascal 64
Permette di usare le più importanti forme sintattiche del linguaggio Turbo Pascal (anche grafiche!) usando un semplice C/64 dotato di drive.
Ideale per studenti.

Disco: L. 19000

Speciale drive
Questo speciale fascicolo costituisce una guida di riferimento per le unità a disco del C64/128.
Comprende anche un velocissimo turbo-disk più la mappa completa della memoria del drive.

Fascicolo + disco: L. 12000

Utility 1
Un dischetto pieno zeppo di programmi speciali per chi opera frequentemente con il drive.

Disco: L. 12000

Utility 2
Seconda raccolta di utility indispensabili per realizzare sofisticate procedure di programmazione.

Disco: L. 15000

Graphic Expander 128
Per usare il C/128 (in modo 128 e su 80 colonne) in modo grafico Hi-res. Aggiunge nuove, potenti istruzioni Basic per disegnare in Hi-Res con la massima velocità in modalità 80 colonne.

Disco: L. 27000

Directory
Come è noto, a partire dal N. 10 di "Software Club" (la rivista su disco per l'utente dei "piccoli" computer Commodore), vengono riportati tutti i listati, in formato C/64-C/128, pubblicati su "Commodore Computer Club".
In precedenza tali listati venivano inseriti, mensilmente, in un dischetto, di nome "Directory", che oltre ai programmi di C.C.C. ospitava decine di altri file tra cui musiche nell'interrupt, giochi, listati inviati dai lettori e altro.
Ogni disco, dal prezzo irrisorio, contiene quindi una vera miniera di software. Ordinando i dischetti di "Directory" si tenga conto che al N. 1 corrispondeva il contenuto del N. 34 di "Commodore Computer Club", al N. 2 il N. 35 e così via.

Ogni dischetto: L. 10000

Super Tot '64
La nuova e completa edizione del programma Tot 13 con tutti i sistemi di riduzione e di condizionamento.
Ampia sezione dedicata alla teoria.

fascicolo + disco: L. 15000

Amiga Totospeed
Finalmente anche per Amiga un programma orientato alla compilazione delle schedine totocalcio.
Fai tredici conil tuo Amiga.

disco: L. 20000

Ecco un "**indice**" degli argomenti (di recente pubblicazione) che riteniamo di maggior interesse per gli utenti della nostra rivista. La suddivisione per **argomenti** faciliterà la ricerca degli articoli; il **nome** di questi non sempre corrisponde a quello originale. Ciò per far meglio comprendere l'argomento trattato. Il numero in neretto corrisponde al **numero del fascicolo** della rivista; in seguito a tale indicazione, separati dal carattere di punto e virgola (;) sono indicati tutti gli articoli, dello stesso tipo, rintracciabili sul fascicolo indicato. Per l'acquisto dei numeri arretrati è necessario riferirsi alle pagine informative poste in fondo al fascicolo che hai tra le mani.

IL MEGLIO DI COMPUTER CLUB

Ms-Dos

80 Ricomincio dal Dos. **81** Districarsi tra i comandi. **82** PcTools V.4 (miniguida). **83** Come usare Word Star (miniguida).**84** Word 5.5 (recens.); Borland Turbo C (recens.). **85** Tre mini file batch; Il file Ansi.Sys.

Pascal Ms-Dos

81 Borland T.Pascal 5.5 (recens.). **82** QuickPascal MicroSoft (recens.). **84** A proposito di variabili.

Pascal Amiga + MS - DOS

86 Una Unit grafica compatibile Kick Pascal e T. Pascal.

Basic Ms-Dos

78 Autocad, scrivilo in Basic. **81** Indovina, indovinello; Da Amiga a Ms-Dos. **82** Agenda automatica per ricordare date importanti. **83** Un generatore di compiti in classe. **84** Simulazione del gioco del 15. **85** Binary game; Briscola. **86** Determinazione del tempo in cui si usa un programma; Come estrarre i nomi dei file Pkzippati.

C Ms-Dos

78 Come eseguire una somma (primi passi). **79** Come animare un cerchio. **81** Microcad. **82** Gestione di file sequenziali e relativi. **83** Gestione del gioco del Lotto (recens.).

Basic + Pascal

79 Due equazioni con tre vestiti. **80** Disegnare in prospettiva.

Basic + Pascal + C

80 Colloquiare con il drive. **82** Barra proporzionale; Girandola. **83** Gestione schermo in modo testo. **85** Invio di file su video e stampante; La divisione infinita. **86** Come esaminare i file dei vostri dischetti.

Assembly 80X86

81 Assembly primi passi; Le istruzioni Mov e Add. **82** Le istruzioni Jmp, Call, Ret. **83** Le istruzioni Dec, Jz, Jnz e Dec. **84** Le istruzioni Jn, Jnz, Int. **85** Introduzione alla grafica.

AmigaDos

75 Assign, Copy, Date, Dir, Install, Path, Search, Sort. **76** Car. spec, Delete, Format, Protect, Rename. **77** Execute, Direttive Batch, If, Skip..lab, Quit. **78** Cd, Ed, Break. **79** Which, Device, Ser, Nil, Raw, Con, Newcon, Par, Prt, Newshell, Newcli. **80** Setmap; Iconx; List. **81** Avail, Join, Alias. **82** Failat, Eval; Un archivio usando i comandi di AmigaDos. **83** Env, Setenv, Getenv. **84** Tre file batch; La memoria RAD. **86** Come spostare le directory; Comandi Escape.

Argomenti di interesse generale

80 Come attuare un collegamento via modem. **81** A proposito di stampanti. **84** Il linguaggio Postscript. **86** Manuale di confusione; In caso di guasto... cambiare computer.

Recensioni di interesse generale

80 Il modem CDC 2400 (hw). **81** Il modem Supramodem 2400 (hw). **82** Amidraw Tablet (hw). **83** DeluxePaint III (sw). **84** I modem US-Robotics; Dos2Dos per trasferire file tra Amiga e Ms-Dos (miniguida). **86** Programmi di conversione grafica tra Amiga ed MS DOS.

Recensioni Amiga

73 Pixmate, s/w grafico. **74** DigiPaint III (s/w); Compressori di file (s/w). **76** The Works parte 1 (sw). **78** The Works Parte 2 (sw). **79** JR-Comm; Stereo professional sample studio (sw-hw); Soundtracker (sw); AC Basic compiler (sw); HiSoft Basic Compiler (sw); GFA Basic Compiler (sw); F-Basic V.2.0 (sw). **80** Hard disk per Amiga (hw); Scheda Ms-Dos per A-500 (hw); Oktalyzer (sw); F-Composer (sw). **81** Drive 5.25 per A-500 (hw); C1-Text V.3 (sw); Movie Setter (sw); Compilatori C (sw). **82** Comic Setter (sw); Trackball Amtrac (hw); Scheda AT per A-500 (hw). **83** Draw4D (sw);AMAS Sampler (hw + sw); Mouse ottico (hw); Hand Scanner JS-105-1M (hw); Amigazzetta 10 (sw). **84** Professional Page (sw); Amos Basic (sw); Audiomaster; Digitalizzatore video (hw + sw); Disk Master (sw). **85** Kick Pascal (s/w); Amigazzetta 11 (s/w); Action Replay 2 (h/w); Master Sound (h/w);Quartet (s/w); Font Maker (s/w). **86** SupraDrive (h/w); Megadrive (h/w); Supra RAM (h/w); Hard Disk Supra per A-2000 (h/w).

Applicazioni per Amiga

73 Come individuare le schermate grafiche presenti nella Ram; Come registrare le nostre schermate grafiche. **74** Come "estrarre" la musica dai videogames. **79** Animare la grafica con Dpaint 3. **85** Tre mini file batch.

AmigaBasic

74 Come realizzare uno scrolling in Basic; Gestione di Bob e Sprite; Generazione di figure di Algomartin. **79** Disegnare meridiani e paralleli. **80** Disegnare in prospettiva; Messaggi cifrati; Domino. **81** Indovina indovinello; Grafici di funzioni tridimensionali; Hard Copy; Determiniamo le formule matematiche "inverse". **82** Un atlante per Amiga. **84** Risposta alla sfida musicale; Risposta alla sfida del tempo. **85** Trasferimento di immagini grafiche da C/64 ad Amiga; Potenze e prodotti con cifre infinite; Binary game; Briscola; File relativi in GfaBasic; Equazioni di terzo grado; Messaggi cifrati usando le matrici. **86** Determinazione di pigreco.

Amiga C

82 Le istruzioni Input/output; Attiviamo uno sprite. **83** La gestione della Ram; **84** La gestione delle stringhe. **85** Vettori, puntatori e matrici **86** Un archivio per dischetti; Come realizzare un ambiente di lavoro.

Amiga Assembly

86 Hard copy; I migliori Assembler per Amiga.

SYSTEMS EDITORIALE PER TE

Disk'o'teca
Grazie a questa nutrita raccolta di brani musicali potrete divertirvi ascoltando i migliori brani prodotti dai vostri beniamini, oltre a una serie di composizioni prodotte "in casa".
In omaggio un bellissimo poster di Sting.
Disco: L.15.000

Assaggio di primavera
Esclusivo!
In un'unica confezione potrete trovare ben due cassette di videogiochi assieme a un comodo e funzionale joystick.
Cassette: L. 15.000

LIBRI TASCABILI

64 programmi per il C/64
Raccolta di programmi (giochi e utilità) semplici da digitare e da usare. Ideale per i principianti. **(126 pag.)**
L. 4800

I miei amici C/16 e Plus/4
Il volumetto, di facile apprendimento, rappresenta un vero e proprio mini-corso di Basic per i due computer Commodore. Numerosi programmi, di immediata digitazione, completano la parte teorica. **(127 pag.)**
L. 7000

62 programmi per C/16, Plus/4
Raccolta di numerosi programmi, molto brevi e semplici da digitare, per conoscere più a fondo il proprio elaboratore.
Ideale per i principianti. **(127 pag.)**
L. 6500

Micro Pascal 64
Descrizione accurata della sintassi usata dal linguaggio Pascal "classico". Completa il volume un programma di emulazione del PL/O sia in formato Microsoft sia in versione C/64 (da chiedere, a parte, su disco). **(125 pag.)**
L. 7000

Dal registratore al Drive
Esame accurato delle istruzioni relative alle due più popolari periferiche del C/64. Diversi programmi applicativi ed esempi d'uso. (94 pag.)
L. 7000

Il linguaggio Pascal
Esame approfondito della sintassi usata nel famoso compilatore. (112 pag.)
L. 5000

Simulazioni e test per la didattica
Raccolta di numerosi programmi che approfondiscono e tendono a completare la trattazione già affrontata sul precedente volume. **(127 pag.)**
L. 7000

Dizionario dell'Informatica
Dizionario inglese-italiano di tutti i termini usati nell'informatica. (Edizione completa). **(385 pag.)**
L. 10000

Word processing: istruzioni per l'uso
Raccolta delle principali istruzioni dei più diffusi programmi di w/p per i sistemi Ms-Dos: Word-Star, Samna, Multimate Advantage, Word 3. **(79 pag.)**
L. 5000

Unix
Un volumetto per saperne di più sul sistema operativo professionale per eccellenza.
Un necessario compendio per l'utente sia avanzato che inesperto **(91 pag.)**
L. 5000

ABBONAMENTO

Computer Club
11 fascicoli: L. 60.000

ARRETRATI

Ciascun numero arretrato di C.C. L. 6.000

Come richiedere i prodotti Systems

Coloro che desiderano procurarsi i prodotti della Systems Editoriale devono inviare, oltre alla cifra risultante dalla somma dei singoli prodotti, L. 3500 per spese di imballo e spedizione, oppure L. 6000 se si desidera la spedizione per mezzo raccomandata.
Le spese di imballo e spedizione sono a carico della Systems se ciascun ordine è pari ad almeno L. 50000.
Per gli ordini, compilare un normale modulo di C/C postale indirizzato a:

C/C Postale N. 37 95 22 07
Systems Editoriale Srl
Via Mosè, 22
20090 Opera (MI)

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento"), non solo il vostro nominativo completo di recapito telefonico, ma anche i prodotti desiderati ed il tipo di spedizione da effettuare.
Per sveltire la procedura di spedizione sarebbe opportuno inviare, a parte, una lettera riassuntiva dell'ordine effettuato, allegando una fotocopia della ricevuta del versamento.
Chi volesse ricevere più celermente la confezione deve inviare la somma richiesta mediante assegno circolare, oppure normale assegno bancario (non trasferibile o barrato due volte) intestato a:

Systems Editoriale
Milano